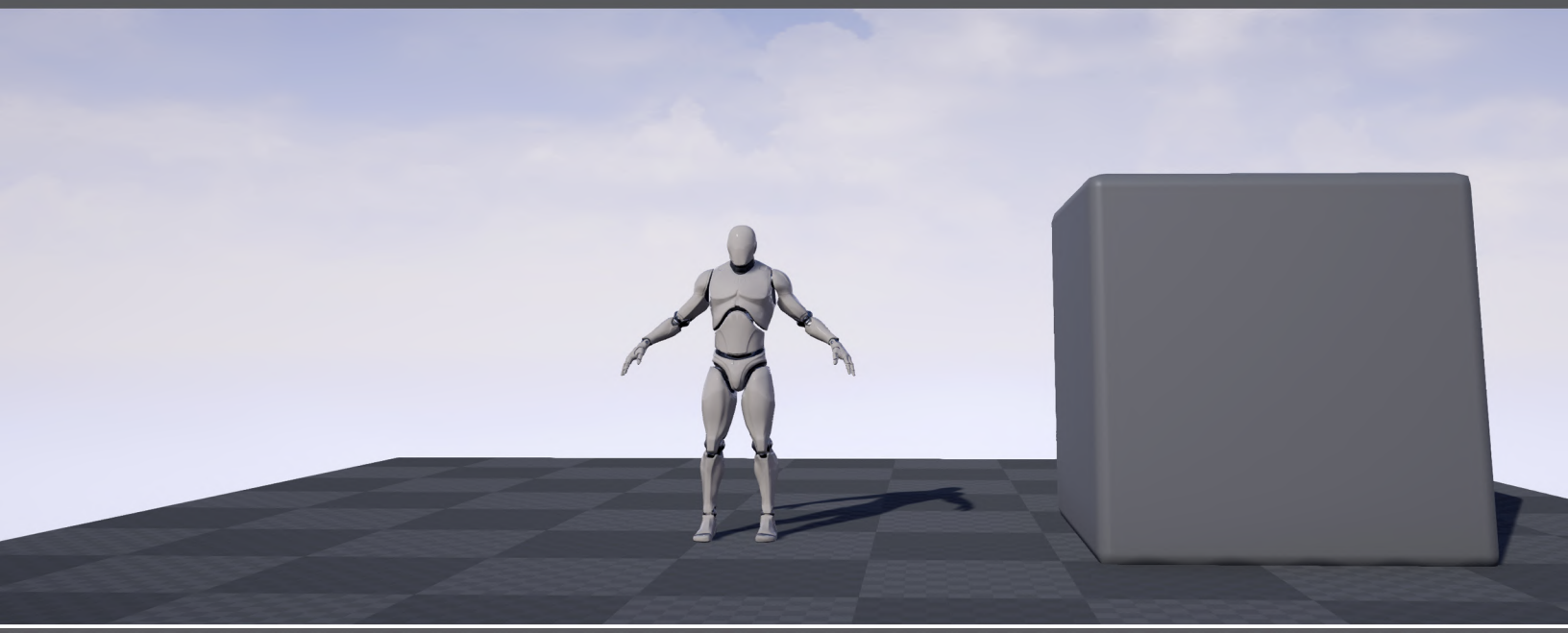
CONTAINS OVER 80+ TECHNIQUES, TIPS AND INSIGHT TO GETTING STARTED WITH UNREAL® ENGINE 4

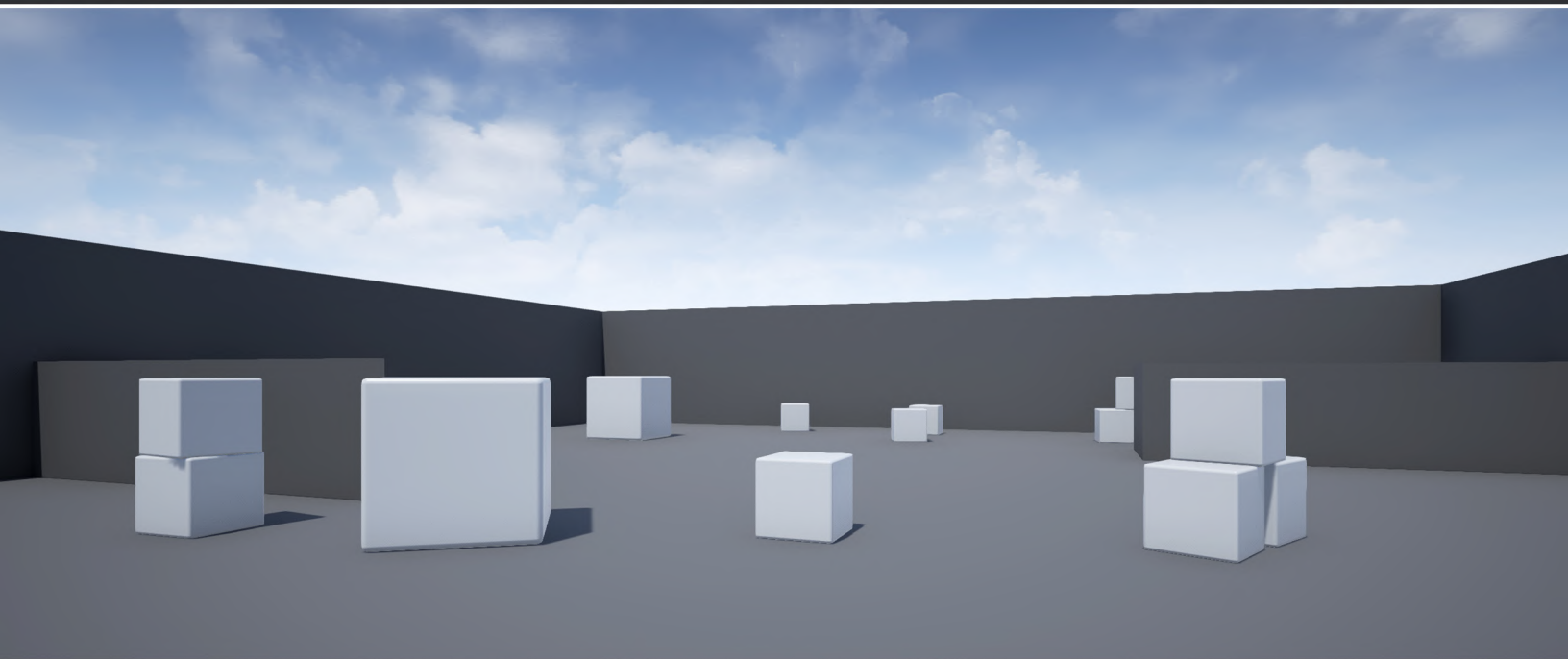


UE4 BEGINNER'S CRASH COURSE

HOW TO START LEARNING AND USING UNREAL® ENGINE 4

FEATURING:

- 16 PRINCIPLES LEARNING UNREAL ENGINE 4
- 27 STEP CRASH-COURSE TO GETTING STARTED WITH UNREAL ENGINE 4
- BEGINNER'S STEP-BY-STEP TO CREATING FIRST PLAYABLE LEVEL IN UE4
- 26 STEPS TO TRANSITION FROM UDK TO UE4



WRITTEN BY: ALEX GALUZIN

V. 1

WORLD OF LEVEL DESIGN.COM

UE4 BEGINNER'S CRASH COURSE

HOW TO START LEARNING AND USING UNREAL® ENGINE 4

WRITTEN BY: ALEX GALUZIN

V. 1

WORLD OF LEVEL DESIGN.COM

“UE4 Beginner’s Crash Course: How to Start Learning and Using Unreal® Engine 4”

Copyright ©2015 World of Level Design™ LLC. All rights reserved.

First Published: September 2015
Version: 1.0
Cover Image: Starter/Example Maps in Unreal® Engine 4
Ebook Created and Published by World of Level Design LLC
www.WorldofLevelDesign.com

No part of this document or the related files may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording, torrent, ftp, website or otherwise). You may not reprint, republish, alter, translate or reproduce this book or any part of it in any way or any language without an explicit written permission of the creator, except in the case of brief quotations embedded in critical articles or reviews; and unless noted otherwise below.

You may NOT upload this book and share it via download in any way.

You may NOT sell or charge money for this book.

You may print this book for your own personal use.

You may share this book via email (attachment) or a private link with another person. As long as it is NOT a public link for everyone to download.

For Schools, Colleges, Universities, or other Educational Institutions:
You may share this ebook freely among students and faculty without needing additional permission. You may share this book with others through print, email, ftp, website or other. This book must stay as is without any alteration or modification.

World of Level Design LLC makes no representation or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose and shall in no event be liable for any loss of profit or any other commercial damage, including but not limited to special, incidental, consequential, or other damages.

World of Level Design LLC has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, World of Level Design LLC cannot guarantee the accuracy of this information.

All trademarks and trade names are properties of their respective owners. All rights reserved.

Epic Games, Inc. Epic, Epic Games, Unreal® Engine, Unreal® Engine 4, Unreal Tournament, Unreal Content and Circle-U Logo are registered trademarks of Epic Games, Inc in the United States of America and elsewhere. All rights reserved.

World of Level Design LLC or “UE4 Beginner’s Crash Course: How to Start Learning and Using Unreal® Engine 4” is NOT affiliated or endorsed by Epic Games or any other mentioned companies in any way.

World of Level Design™ is a trademark of World of Level Design LLC and Alex Galuzin.

You may contact World of Level Design at: contact@worldofleveldesign.com

TABLE OF CONTENT

Section 1: 16 Principles to Learning Unreal Engine 4

- Get Unreal Engine 4
- What Do You Want From UE4?
- Do Not Create A Game Or Custom Environment As First Project
- Steps Of How Games Are Created In UE4
- Steps Of How Environments Are Created In UE4
- Your Goal As A Complete Beginner
- Set At Least 1 Hour Each Day To Learn UE4
- Start With UE4 Project Game Templates
- First Recommended Project
- Set A Small Project To Learn A Skill
- Focus On Learning Skills That You Need Right Now
- Second Recommended Project
- Third Recommended Project
- Fourth Recommended Project
- Fifth Recommended Project
- Ask For Help

Section 2: 27 Step Crash Course to Unreal Engine 4

- Download And Install
- Create Your First Project And Launch The Editor
- Use Starter Content
- First Time Inside The Editor
- Creating, Saving And Opening Levels/Maps
- How To Work With Viewports
- Viewport Navigation
- Working With Actors/Objects (Move, Rotate, Scale)
- Local And World Coordinate Systems
- Details Panel/Object Properties
- Content Browser
- Inserting Static Meshes
- Static Mesh Editor
- Static Mesh Editor vs Static Mesh Details Settings
- Use Grid Snapping
- BSP Brushes
- Adding And Subtracting BSP Brushes
- Using Geometry Editing Mode
- BSP vs Static Meshes
- Player Scale And Proportion
- Inserting A Player Start

- Adding Directional Light (Sun Light)
- Adding Point Light Or Spot Light (Interior Lights)
- World Outliner
- Play Test Your Level
- Build All
- Marketplace Content and Learn Section

Section 3: Beginner Step-by-Step Basics to Creating Your First Level

- Setting Up
- Launch The Editor
- Start A New Blank Level
- Ground Plane
- Character Reference Scale
- Directional Light
- Atmospheric Fog
- Insert Static Mesh
- Skylight
- Player Start
- Save Your Level
- Test Your Level
- Build All
- What's Next

Section 4: Transition from UDK to UE4

- Free Unreal Engine 4
- Unreal Engine/Epic Games Launcher
- Project Management
- Game Templates
- New Level Templates
- Keyboard Shortcuts
- Tabs And Fluid Interface
- Viewport Navigation In UE4
- Modes Panel Class Viewer
- Developer Tools
- Content Browser
- Working With Objects
- Details Panel
- Show All Advanced Details Property
- Scale
- Grid Setting Size
- World Outliner
- World Settings

- BSP Brushes
- PBR Material Creation
- Lighting
- Blueprint Replaces Kismet
- Importing And Storing Custom Assets
- Project Settings
- Current Project In UDK To UE4

FROM WORLD OF LEVEL DESIGN.COM

Hi,

Alex here,

I want to thank you for downloading this guide from World of Level Design.com

I love level design and game environment art. Most of my time consists of learning, testing and experimenting inside various game engines. I then take what I learn and create tutorials, videos, and guides. I do all the work, so you don't have to and so you can focus on learning what is necessary.

Below, I have listed how to keep in touch with WoLD to get more tutorials, guides, videos, tips and insight so you can become THE BEST level designer and game environment artist.

WoLD Website (MAIN SOURCE):
http://www.worldofleveldesign.com/

WoLD Insider Newsletter and Updates:
If you haven't signed up for a free "WoLD Insider Newsletter and Updates" visit here and get a couple of free guides for signing up:
http://www.worldofleveldesign.com/wold-insider/

WoLD Facebook:
https://www.facebook.com/worldofleveldesign

WoLD Twitter Channel:
https://twitter.com/GameLevelDesign

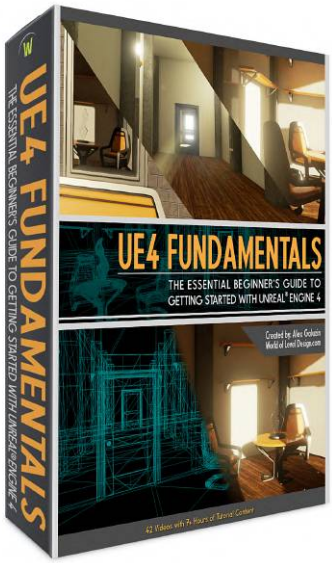
WoLD YouTube Channel:
https://www.youtube.com/WorldofLevelDesign

WoLD Google+ Channel:
https://plus.google.com/+WorldofLevelDesign/

PREMIUM VIDEO TUTORIAL GUIDES FOR UE4

For more in-depth tutorials on how to use and master Unreal Engine 4, see these premium video tutorial guides:

"UE4 FUNDAMENTALS: Essential Beginner's Guide to Getting Started with Unreal Engine 4"



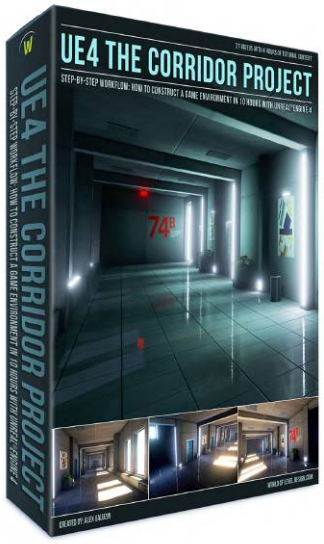
The essential beginner's guide to learning and using Unreal Engine 4. This is a complete beginner's guide to getting started with UE4.

In this guide you will get:

- 42 videos with 7 hours of tutorial content
- The Essential Beginner's Guide to Getting Started with Unreal Engine 4
- Instant Digital Download

Click here for more info...

"UE4 THE CORRIDOR PROJECT: Step-by-Step Workflow to Construct a Game Environment in Less Than 10 Hours with Unreal Engine 4"



UE4 The Corridor Project is an intermediate tutorial guide focused on constructing a game environment with provided custom Static Meshes. It's in-depth guide for putting together an environment from start-to-finish.

In this guide you will get:

- 21 videos and 4 hours of tutorial content
- Step-by-Step Workflow to Construct a Game Environment in Less Than 10 Hours with Unreal Engine 4
- Instant Digital Download

Click here for more info...

INTRODUCTION AND ABOUT THIS GUIDE

INTRODUCTION

Getting started with a brand new game engine is always intimidating and often requires a lot of time to learn.

But it doesn't have to.

I created this UE4 Crash Course guide to help you speed up the learning process with Unreal Engine 4 and make you use it as quickly as possible.

In this guide you will learn:

- 16 Principles For Learning Unreal Engine 4
- 27 Steps To Getting Started With Unreal Engine 4 Editor
- Step by Step Guide to Creating Your First Playable Level
- 26 Steps to Transition from UDK to UE4

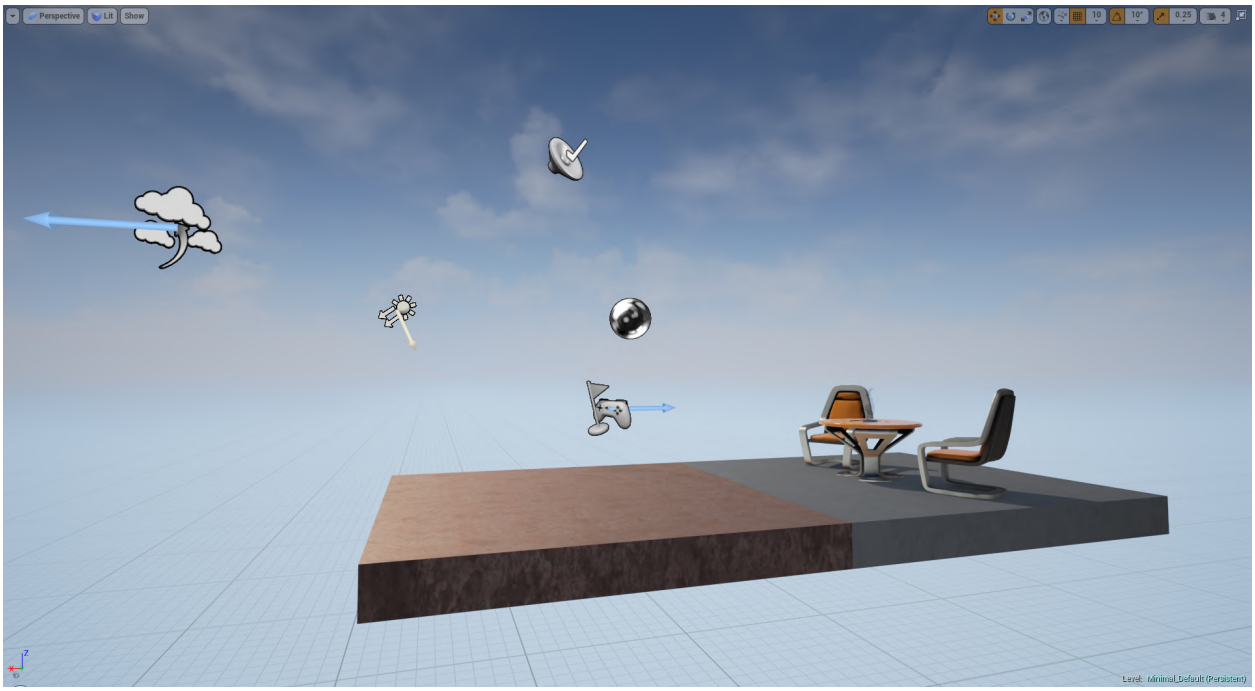
TESTIMONIALS AND FEEDBACK

If you enjoyed the content in this ebook, please submit a short testimonial on what you enjoyed and learned from this guide to support@worldofleveldesign.com or post it to World of Level Design Facebook and Twitter.

Please submit any mistakes and typographical errors you find to support@worldofleveldesign.com

Thank you for reading this guide and spreading the word about World of Level Design.com

SECTION 1:
16 PRINCIPLES LEARNING UNREAL ENGINE 4



Learning a new game engine as a complete beginner is very intimidating. There are a lot of tutorials, documentation and advice already out but how do you start and proceed with learning Unreal Engine 4 is unclear. You get pulled into many different directions and end up confused and overwhelmed.

I have spent a lot of time deconstructing what it takes to learn a game engine from scratch. What it is that you should focus on first and what you should avoid until later.

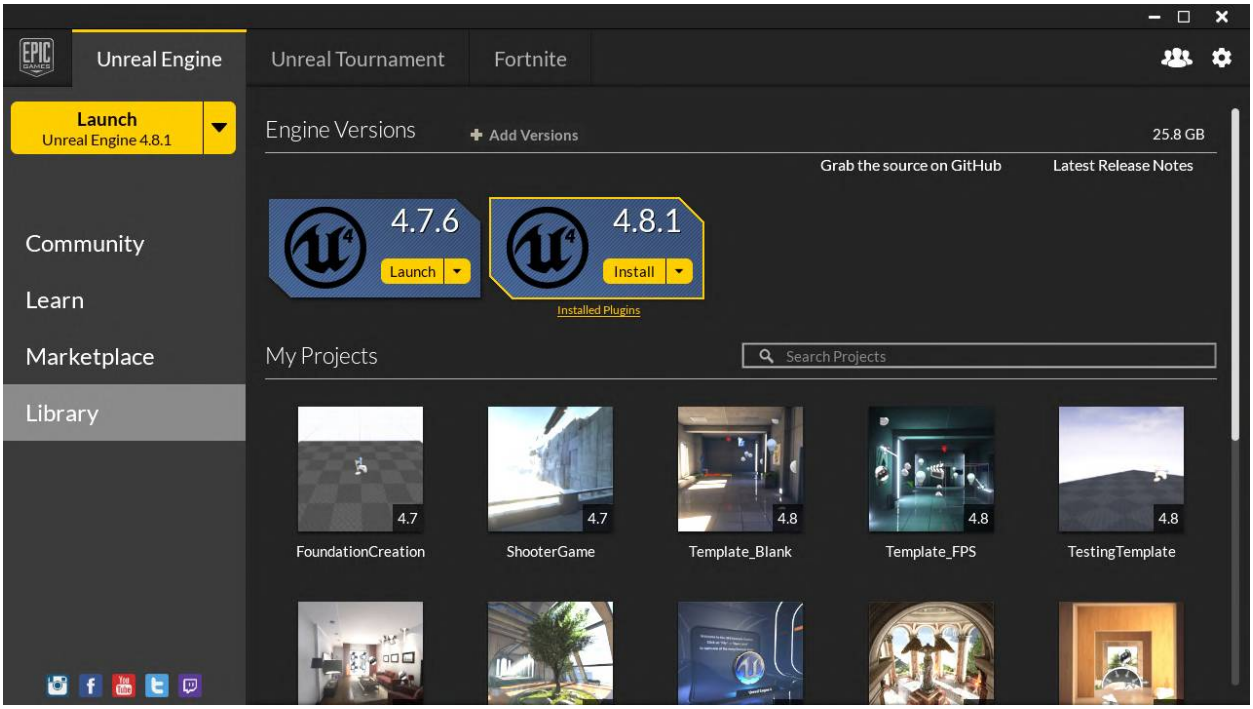
Unreal Engine 4 is a complete game engine. It is an extremely deep and complex piece of software that can be used to create variety of games, environments, cinematic and visualizations.

So how do you learn UE4 from scratch?

In this section you will learn 16 principles/guidelines to follow that will give you direction to learning Unreal Engine 4 easier.

1. GET UNREAL ENGINE 4

First, download and install Unreal Engine 4. It is free. [Follow this tutorial on how to download and install]. Make sure that your computer hardware is up to specifications.



- Recommended Specs are:
- Operating System: Windows 7/8 64-bit
- Processor: Quad-core Intel or AMD, 2.5 GHz or faster
- Memory: 8 GB RAM
- Video Card: DirectX 11 compatible graphics card
- DirectX Version: DirectX End-User Runtimes (June 2010)

https://docs.unrealengine.com/latest/INT/GettingStarted/RecommendedSpecifications/

2. WHAT DO YOU WANT FROM UE4?

Most important part in learning any game engine is to know what you want from it. What is the outcome you are after? Why do you want to learn and use UE4?

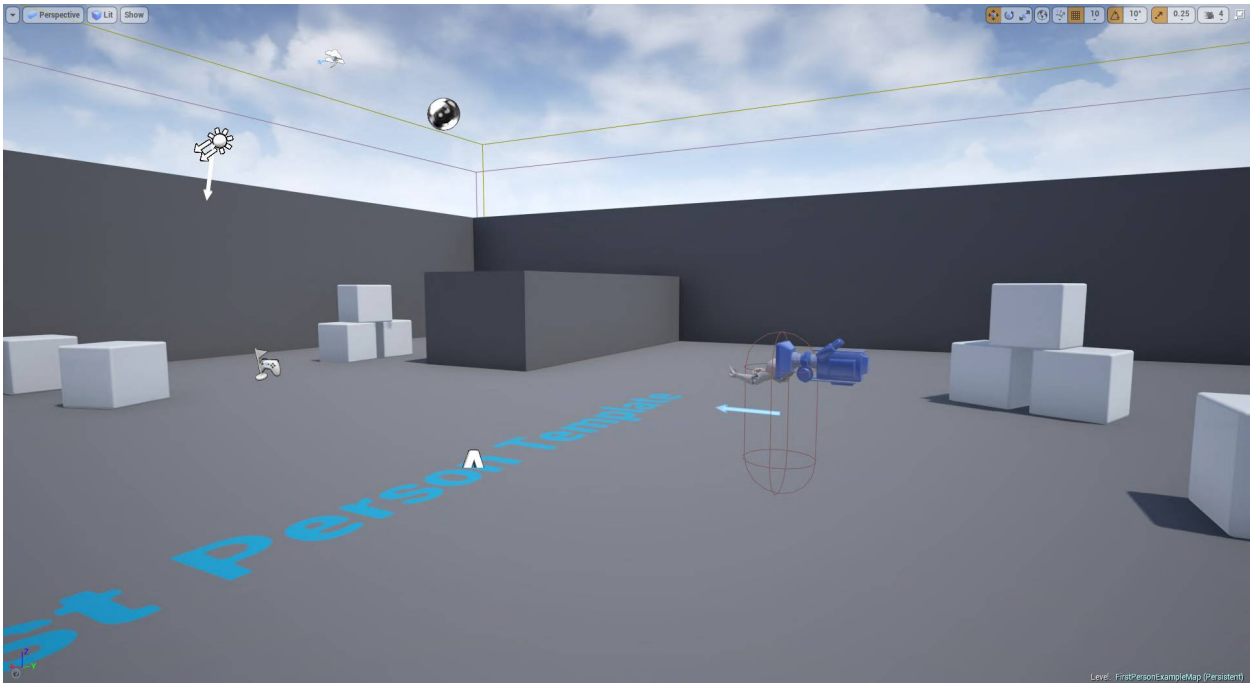
Is it:

- to create a game?
- to use it for arch viz?
- to use it for game environment art?
- to create playable levels for a game?
- to create cinematics?

You might be thinking, all of the above - and that is fine. But as a complete beginner learning Unreal Engine 4, you want to narrow this down to an essential single focused outcome. You can expand on your knowledge later once you have become proficient with the engine. But this early in the learning process, decide on a single outcome.

I've made this mistake too many times as I wanted to a make full open world environments when I haven't even opened the software.

My primary focus has always been to create game environments and level designs. So everything I learn about a game engine tends to be very focused on helping me become a better level designer and environment artist.



I cannot stress this principle enough:

- Why do you want to learn UE4?
- What outcome are you after?

Decide on one single aspect of why you want to learn Unreal Engine 4, because this will dictate what you should and should not focus on.

3. DO NOT CREATE A GAME OR CUSTOM ENVIRONMENT AS YOUR FIRST PROJECT

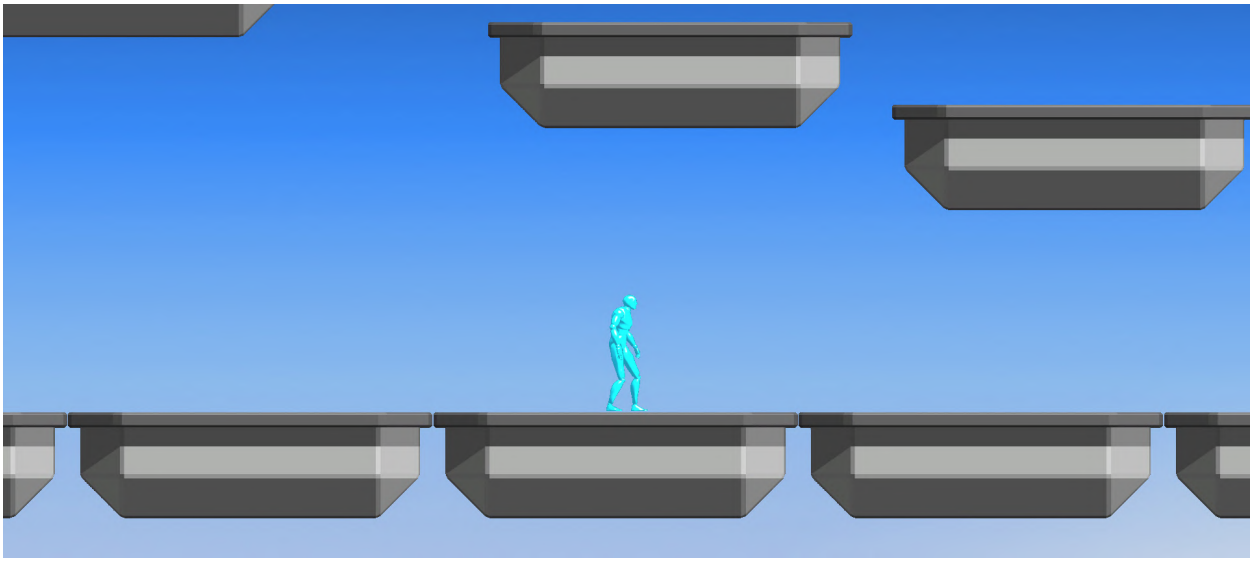
I know this is going to be very tempting. Especially with all the amazing work you see being done with UE4.

Creating a game or a custom game environment contain many different disciplines and skills. Each skill will often require its own time to learn and develop outside Unreal Engine 4. For example 3d modeling, texturing or C++.

So keep the big idea in your mind as a goal you want to do, just resist creating something that complicated for your first project.

Later in this article I will outline 5 recommended projects for you to start with. You can infuse each of these projects with your own ideas. But as you start out, don't create a game or a custom game environment as your very first work inside Unreal Engine 4.

4. HOW GAMES ARE CREATED IN UE4



As you learn UE4 or any other game engine, it helps to have an overview of how games and game environments are created. It helps to have a top down view of the entire process.

So, here is a simple overview of how games are created. This isn't a definitive list to follow but only to be used an example to demonstrate various steps of the process.

- Idea
- Concept Art: visual style of the game (drawings, digital paintings, photo references)
- Gameplay Concept: how will the game play; gameplay mechanics; game design document
- Gameplay Prototyping (Blueprint or C++): simple playable game with core mechanics integrated into it and placeholder items
- Continue to Refine and Finalize Gameplay Mechanics
- 3D Static Mesh Creation (3d models, UVing, texturing, Materials, export/import
- Animations
- Effects
- Level Design (BSP prototyping, gameplay mechanics, scripting, AI, play testing, detailing with Static Meshes, lighting)
- Optimizing/Performance
- Packaging
- Release

Many of these steps would overlap during the production. Also testing and bug fixing would be a constant process step throughout the entire production.

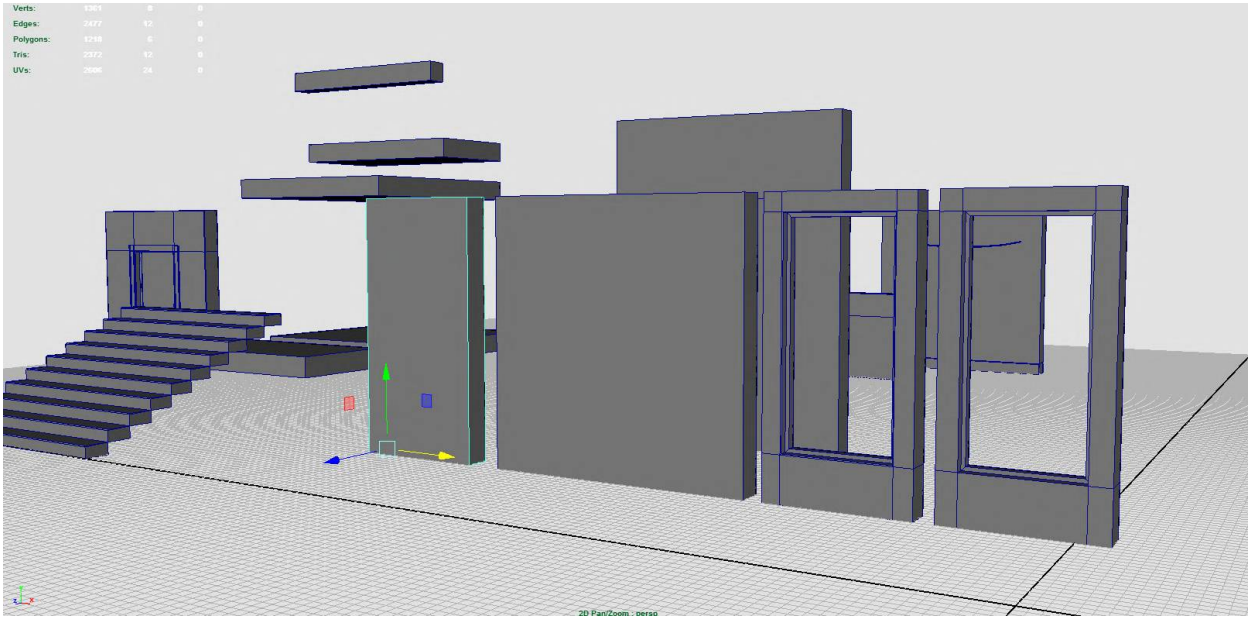
As you can see, there are a lot of disciples and skills that are involved. As a beginner with UE4, you will be quickly overwhelmed at attempting to create your own game without learning the fundamentals of the editor.

Now, let's cover what it takes to create a game environment with Unreal Engine 4.

5. HOW ENVIRONMENTS ARE CREATED IN UE4

Just as with creating a game, creating a custom 3d game environments contain a lot of different skills sets.

Here is an overview of how custom game environment are created. This is not THE pipeline; but one of many. Depending on the environment some steps may not be necessary.



- Create 3d model: high poly mesh first or low poly mesh first
- UV low poly for textures
- UV low poly for lightmaps
- Optimize low poly (reduce poly count, set up smoothing groups)
- Bake high poly to low poly
- Bake other textures if needed
- Create LOD (level of detail)
- Create custom collisions
- Export from 3d modeling package
- Import Static Meshes into UE4
- Create/paint textures
- Export/Import all textures into UE4
- Create materials in UE4
- Finalize all Static Meshes and Materials in UE4
- Prototype/Block In the level with BSP brushes in UE4
- Set up Blueprint for gameplay mechanics (if required)
- Construct the game environment inside UE4 with final Static Meshes replacing BSP brushes

Many of these steps would overlap during the production. Also testing, re-baking and exporting/importing would be a constant process step throughout the entire production.

6. YOUR GOAL AS A COMPLETE BEGINNER

Your first goal as a complete beginner is to learn the most commonly used functions and techniques. These are the fundamentals of UE4 and becoming an expert user. This doesn't sound as exciting as creating a full game or open world environment but it is what will make that possible.

These beginner techniques and functions include knowing how-to:

- Create and manage your projects
- Viewport navigation
- Use the Content Browser
- Place and work with objects and static meshes
- Use BSP to block-in a level layout
- Use static meshes to construct environments
- Inserting and using lights such as Directional Light, Spotlights and Pointlights
- Manage your scene
- Inserting required/needed level objects such as Lightmass Importance Volume, Player Start, Fog etc
- Building to scale and proportion
- Building All function
- Play testing from the editor
- Various shortcuts to speed up all of these processes until they are automatic

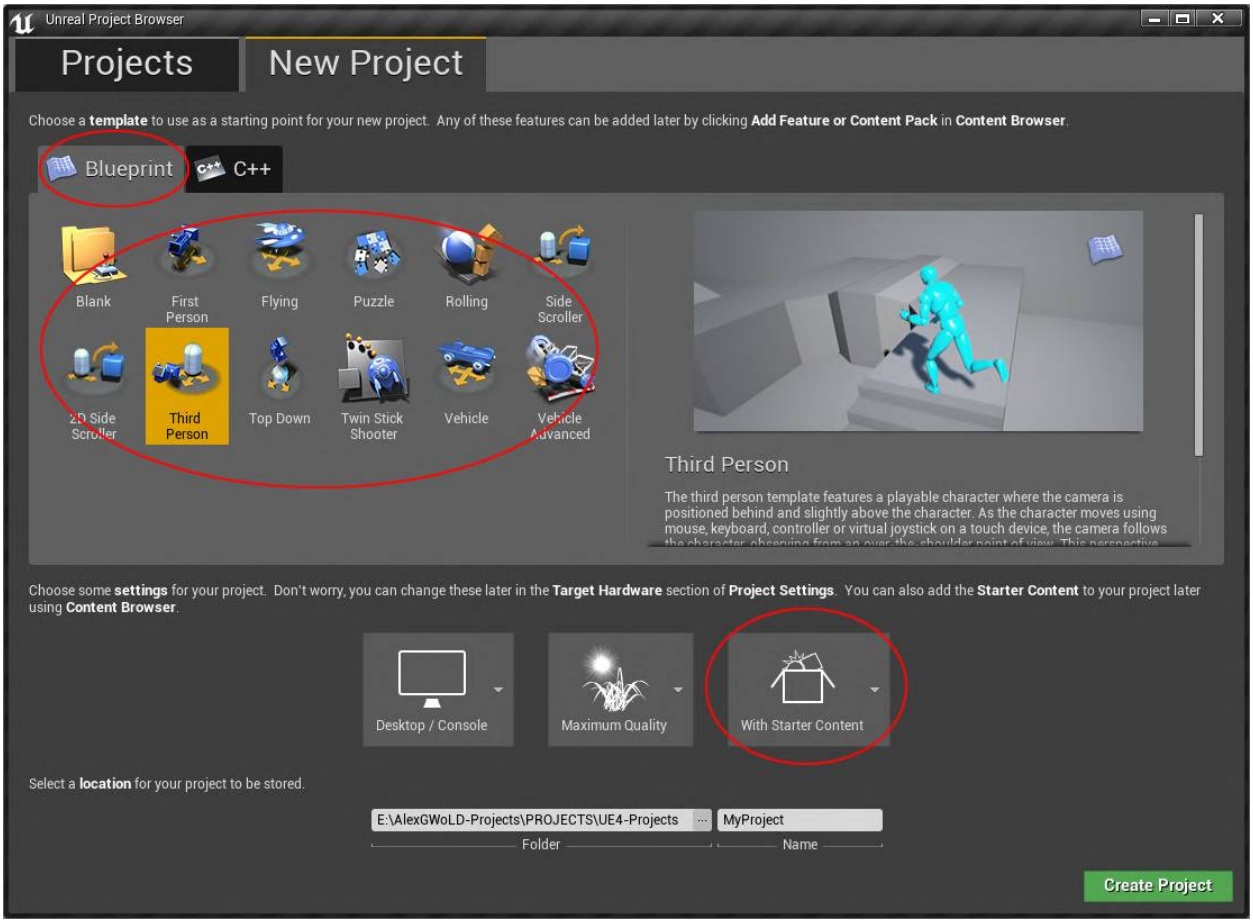
These are some of the essential basics you should know to become proficient with UE4.

7. SET ASIDE AT LEAST 1 HOUR EACH DAY FOR THE FIRST 14-28 DAYS

Set up a system of learning UE4 on daily basis. For the next 2-4 weeks, work for at least one hour per day, every day learning the engine.

One hour should be enough to start with. You can do more, but one hour should be your minimum.

8. START WITH UE4 PROJECT GAME TEMPLATES


Unreal Project Browser
Projects
New Project
Choose a template to use as a starting point for your new project. Any of these features can be added later by clicking Add Feature or Content Pack in Content Browser.
Blueprint
C++
Blank
First Person
Flying
Puzzle
Rolling
Side Scroller
2D Side Scroller
Third Person
Top Down
Twin Stick Shooter
Vehicle
Vehicle Advanced
Third Person
The third person template features a playable character where the camera is positioned behind and slightly above the character. As the character moves using mouse, keyboard, controller or virtual joystick on a touch device, the camera follows
Choose some settings for your project. Don't worry, you can change these later in the Target Hardware section of Project Settings. You can also add the Starter Content to your project later using Content Browser.
Desktop / Console
Maximum Quality
With Starter Content
Select a location for your project to be stored.
E:\AlexGWoLD-Projects\PROJECTS\UE4-Projects
MyProject
Folder
Name
Create Project

Creating your first Unreal Engine 4 project, make sure to use these 3 things:

- First, use one of the available game templates. There are many to choose from, such as First Person Shooter, Third Person Shooter, Top Down or Side Scroller template. Using these will give you basic gameplay mechanics with your project. You could reverse engineer them for learning and build on the existing functionality by adding your own changes.
- Second, select to use Blueprint. Blueprint is a visual scripting language inside UE4 and does not require you to have any C++ programming knowledge in order to incorporate custom behavior in your project. Blueprints are extremely powerful and you can use it to create your own games and game types without having to learn C++ just yet.
- Third, include Starter Content with you project. This will give you a few assets (Static Meshes, textures and materials) to use with your work.

Using these 3 elements will set you up on the right track.

So, what should be your very first project with UE4?

9. FIRST RECOMMENDED PROJECT: SMALL ENVIRONMENT WITH STARTER CONTENT

The key to your first UE4 project is to keep it very small. Your goal is to learn UE4 functionality, the fundamentals and principles of using the game engine.

So, the first recommended project for all complete beginners to start with is to create a small environment such a single room, hallway or exterior area with Starter Content.

Small environment will keep the scale of your project down to a reasonable level while you learn UE4, and most importantly - you will finish your project. Pick a simple idea that supports Starter Content assets and create a simple environment while focusing on learning the fundamentals of UE4.

Check out this first recommended project as a **full tutorials guide series "UE4 Fundamentals"**, which will take you through all the steps you need to know to learn Unreal Engine 4.

10. SET A SMALL PROJECT TO LEARN A SKILL

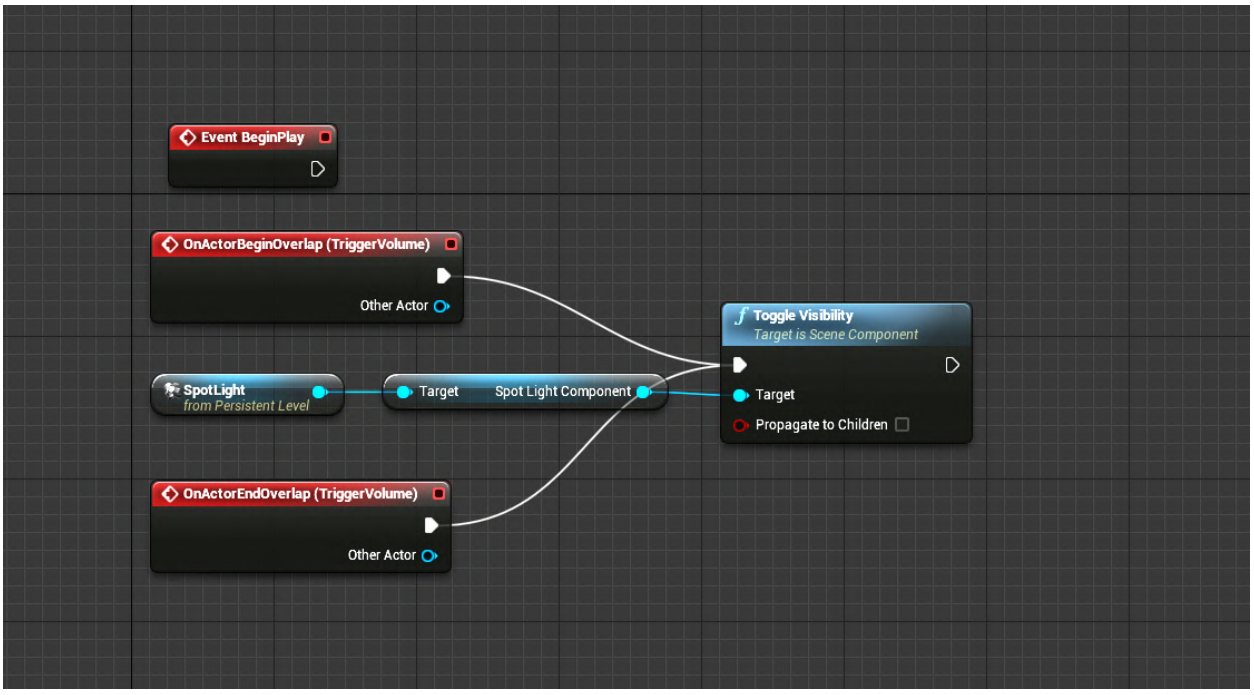
The key is to create a small project that incorporates you learning a new skill. The first recommended project does this. It set up boundaries of self-imposed limitation with Starter Content and creating a small basic environment while learning the fundamentals of UE4.

Each project that is recommended in this article uses the same principle. It defines a small project that can be completed in short amount of time to acquire a certain skill set.

For any new additional skill that you want to learn, follow these steps:

- Set up a small project
- Define a clear outcome that you want to achieve at the end of the project - what is the result you are after
- Work for at least 1 hour per day to complete this small project
- Have something to show at the end, such as a screenshot or a video of what you have done

As an example if you want to learn Blueprint, you would set up a small project and define a clear outcome. This could be “Complete 3 Blueprint Scenarios” - create intractable door, switchable/toggleable light and set up player camera for top down gameplay.



You are training yourself to finish what you start. Once you take on a larger project, you've built momentum and experience for completing. At that point you would just be increasing the scale of the environments you create and complexity of games you design.

11. FOCUS ON LEARNING SKILLS THAT YOU NEED RIGHT NOW

You want to learn only what you need right now, to finish a project. Do not try to learn everything at once. This is why you don't want to create a game or custom game environment as your first project - too many skills and steps all at once.

The key to learning any game engine, especially UE4 is to set self-imposed limitations.

In order to focus on learning a specific skill it requires you to break learning into skill chunks.

Just as you read Principle #4 and #5 of what it takes to create a game or a custom game environment, you would break the large project into a smaller projects, sessions or skill chunk.

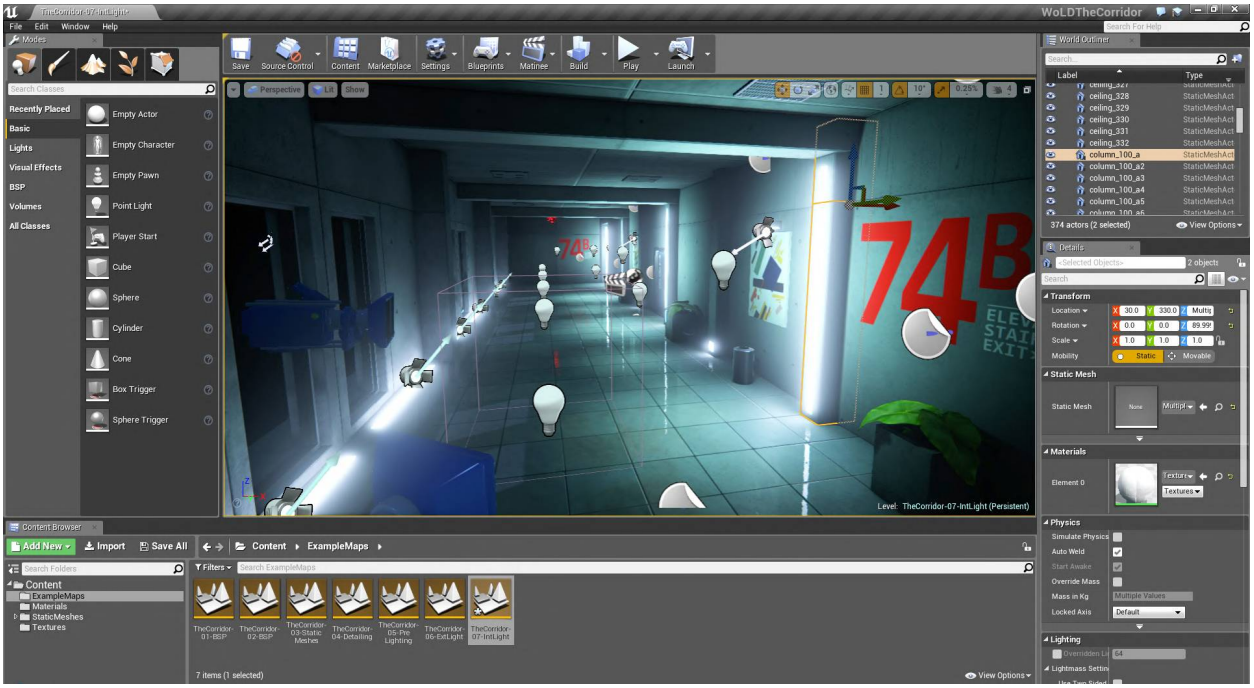
Do not read all the books, tutorials and watch all the videos you can find and then sit down and start working on your project. Don't take all of the UE4 information at once. Focus on 1-2 skill areas to accomplish a project.

If you run into a problem, watch a video or read a tutorial - fix the problem and continue with your project. Focus on 1-2 skill chunks at a time.

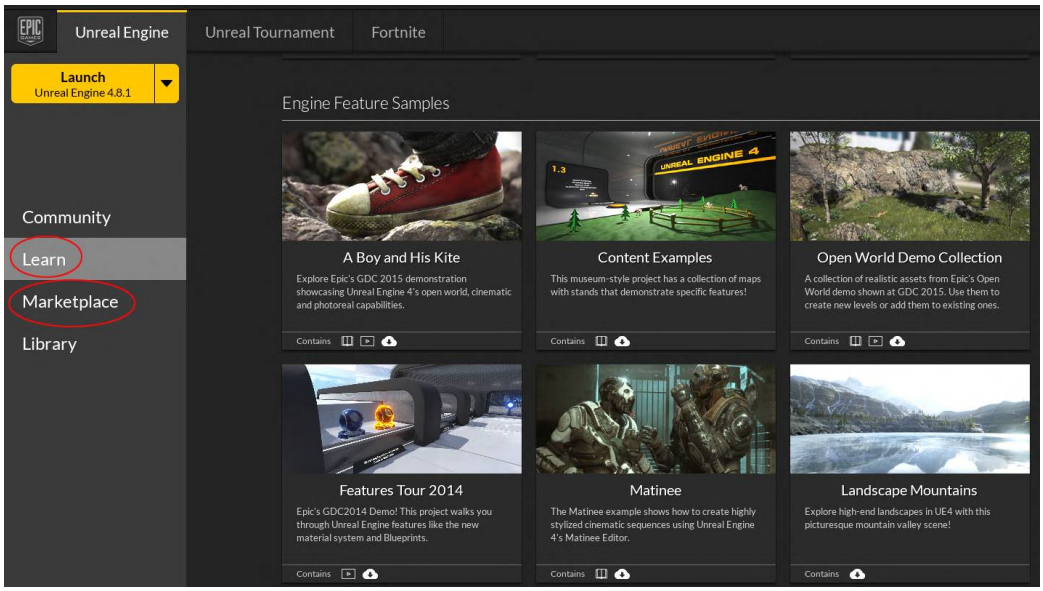
Here is a quick overview of focusing on what you need right now:

- Set a project
- Find tutorials
- Follow each tutorial and put what you learn into practice right away
- Learn that skill
- Have a result to show
- Move on to the next skill

12. SECOND RECOMMENDED PROJECT: ENVIRONMENT WITH CUSTOM CONTENT



Once you are proficient with using UE4 and created your first Starter Content environment - next project is to create a slightly larger environment with a set of new custom static meshes. These custom static meshes should be either something you already have or downloaded from Marketplace/Learn sections.


Unreal Engine
Unreal Tournament
Fortnite
Launch
Unreal Engine 4.8.1
Community
Learn
Marketplace
Library
Engine Feature Samples
A Boy and His Kite
Explore Epic's GDC 2015 demonstration showcasing Unreal Engine 4's open world, cinematic and photoreal capabilities.
Content Examples
This museum-style project has a collection of maps with stands that demonstrate specific features!
Open World Demo Collection
A collection of realistic assets from Epic's Open World demo shown at GDC 2015. Use them to create new levels or add them to existing ones.
Features Tour 2014
Epic's GDC2014 Demo! This project walks you through Unreal Engine features like the new material system and Blueprints.
Matinee
The Matinee example shows how to create highly stylized cinematic sequences using Unreal Engine 4's Matinee Editor.
Landscape Mountains
Explore high-end landscapes in UE4 with this picturesque mountain valley scene!
Contains

Don't create your own custom Static Meshes yet.

You want to expand your UE4 knowledge and experience beyond the beginner by forcing yourself to create larger and more unique environment while using a set of Static Meshes already created.

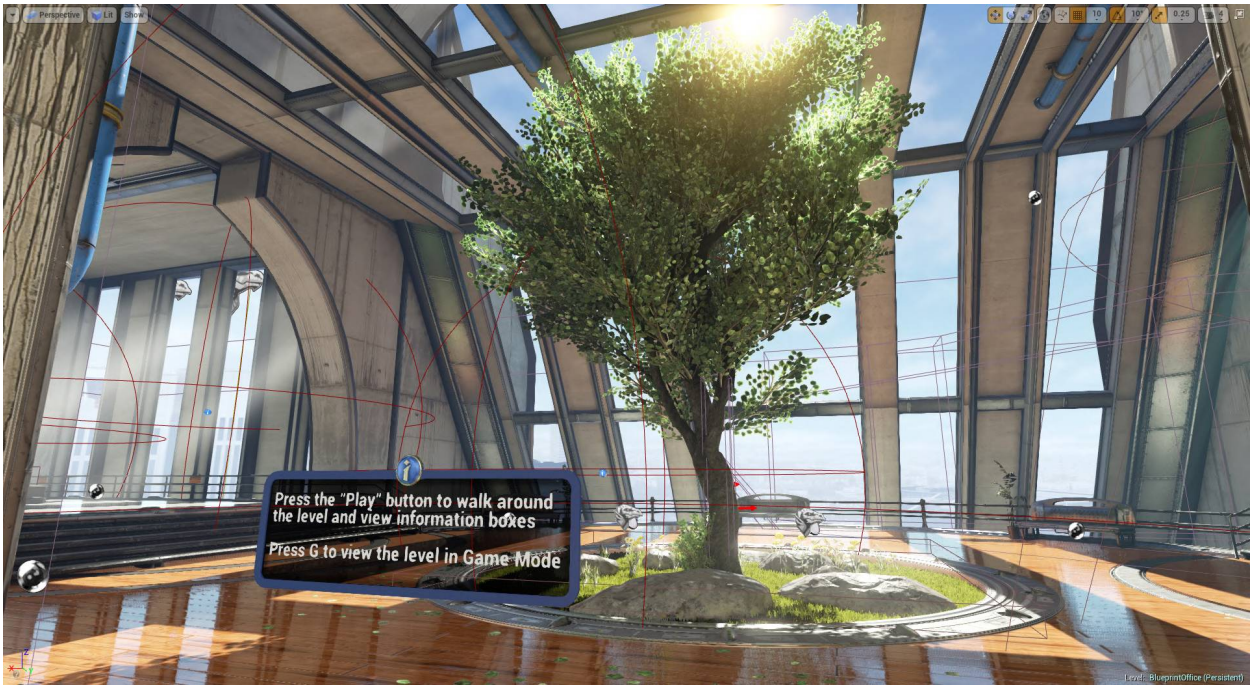
The focus is to construct a game environment with a custom set of Meshes. You are free to explore any idea you want, as long as you find a set of Static Meshes that support it.

Take a look at **"UE4 The Corridor Project" tutorial series**, which focuses on using a set of custom Static Meshes to construct a game environment.

13. THIRD RECOMMENDED PROJECT: LEARN BLUEPRINT TO CREATE BASIC LEVEL INTERACTION

Your third project is to start learning Blueprint.

Blueprint is the visual scripting language inside UE4. So instead of having to learn programming in C++, Blueprint will do the work for you. This means you can script most of the gameplay and game mechanics inside the editor without having to write and compile code yourself. It is possible to create a game using nothing else but Blueprint.



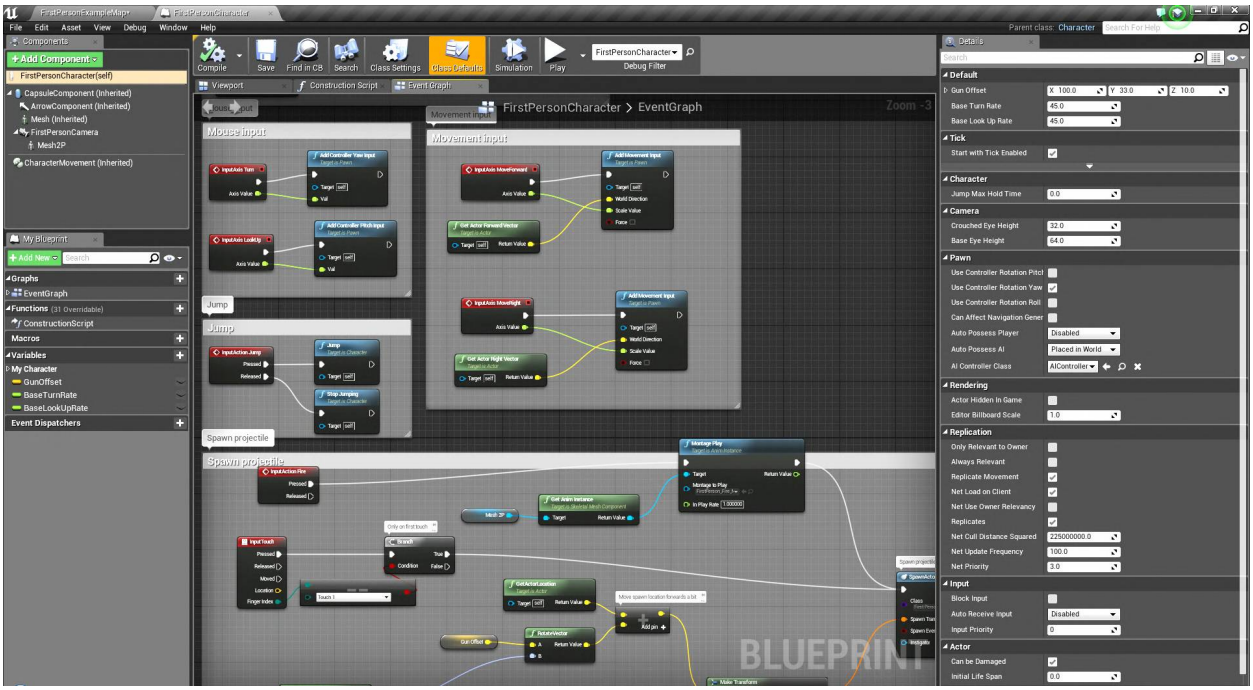
Learn basic functionality with Blueprint to create interactive gameplay elements. Moving platforms, open/close doors, light switch, cause and effect - pick something up and then something happens.

Learning the basics of Blueprint, such as:

- Opening/Closing a door
- Toggleable light
- Moving platform
- Trigger an event (such as a door to open, light to turn on, object to move)
- Activating certain events or actions on level load or on player spawn
- Adding interaction via a button or text on screen

Blueprint is extremely powerful and it is essential to learn it, especially if you are a level designer or game designer.

14. FOURTH RECOMMENDED PROJECT: DECONSTRUCT AND RE-CREATE ANY BLUEPRINT GAME TEMPLATE



Your fourth project is to deconstruct one of UE4 game template such as first person shooter, third person, top-down or any other. Reverse engineer how Epic used Blueprint to create the game template then re-create it yourself.

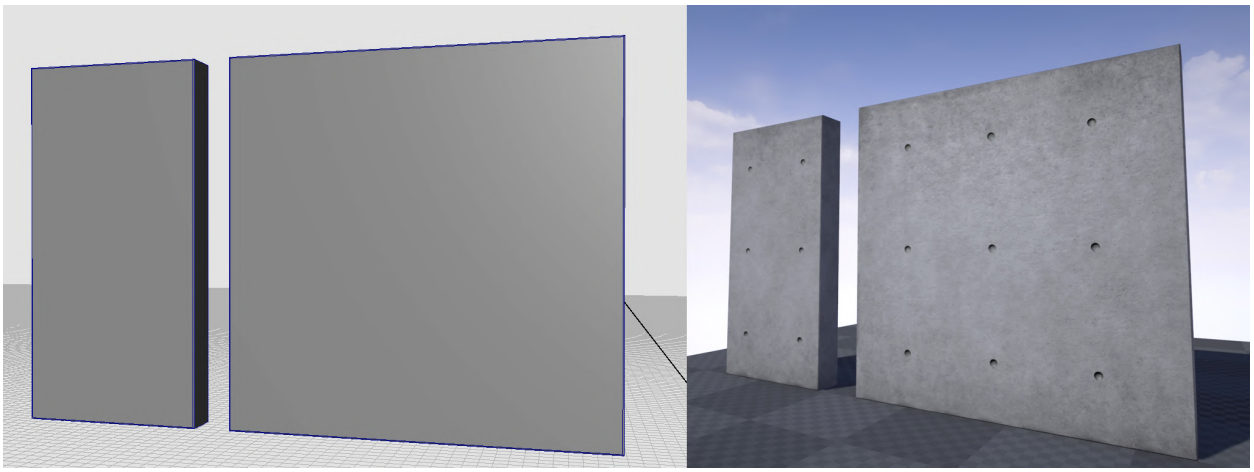
Pick any of the Blueprint Game templates; learn, take notes and reference it in order to learn as much as you can - then start a new Blank Template and re-create the same functionality.

You can go back and re-open Epic's Game Blueprint template to double check at any time. But don't actually copy/paste anything from the game template, insert each node yourself and re-create it manually.

The Blueprint game template you choose should be similar to the type of game you want to create. So if it is a first person shooter you are after, then re-create the FPS Game Template.

15. FIFTH RECOMMENDED PROJECT: MODEL YOUR OWN STATIC MESHES AND IMPORT THEM INTO UE4

Fifth recommended project is to learn what it to create your own custom Static Meshes and import these models into UE4.



This is where you would need to learn 3d modeling software such as Maya LT, Maya, 3dsMax, Modo or Blender. Then, model, UV, texture, lightmap, create materials and export/import everything into UE4 to construct the environment.

If you want to be a game environment artist, then you have to learn 3d modeling package. If you are a level designer, a programmer or texture artist then learning 3d software is optional. In many game studios and game design teams you would have a specific person dedicated to creating 3d environment art. But depending on the size of a studio/team, many non-

environment artists are expected to know how to create and texture their own models.

My own personal take on this topic is if you don't want to become a 3d environment artist, you should still learn to use 3d modeling package. You don't have to master it but knowing how is a great skill to have. There is only upside to you knowing how to use 3d software. The time it takes to learn it is an investment and will make you more valuable as an artist.

16. ASK FOR HELP

Throughout all of these projects, do not hesitate to ask for help. But if you get stuck, spend time figuring it out first. Exhausted your own problem solving skills before turning for help.

Here are places to visit for answers:

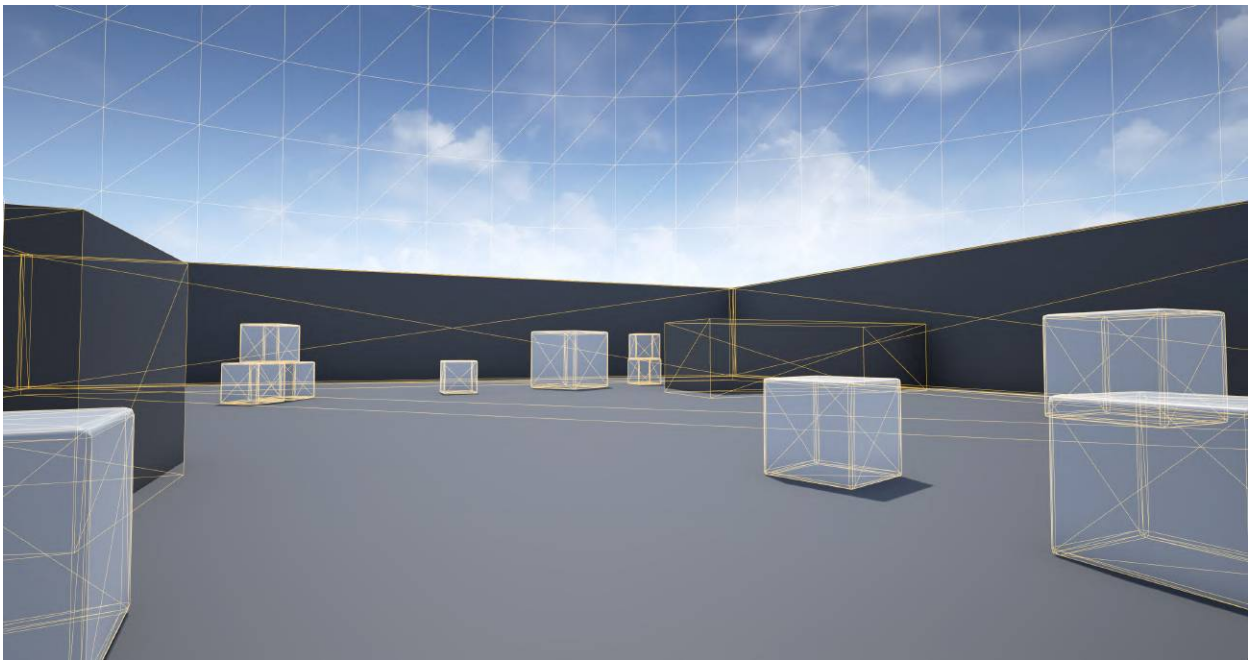
- **WoLD UE4 Tutorials:** http://www.worldofleveldesign.com/categories/cat_ue4.php
- **Unreal Engine Answer Hub:** https://answers.unrealengine.com/
- **UE4 Forums:** https://forums.unrealengine.com/
- **Unreal Engine YouTube Channel:** https://www.youtube.com/user/UnrealDevelopmentKit/
- **Google:** https://www.google.com/
- **YouTube:** https://www.youtube.com/

WHAT IS NEXT

After you accomplished five recommended projects, you are set to explore UE4 in more advanced topics.

You are now in a great position to explore in starting your own game, creating your own custom environment and go deeper into more advanced UE4 topics.

SECTION 2:
27 STEPS TO GETTING STARTED WITH UE4



Unreal Engine 4 is a complete game engine; it is very expansive and can be overwhelming to a complete beginner.

Common question is, “Where do I even begin?”

The following section is an overview of what you need to know right now to get started with UE4. It is perfect for those who have never used and are very intimated by Unreal Engine - but need a quick to the point overview to begin using this game engine.

If you are interested in more methodical and step-by-step approach to learning Unreal Engine 4, then **I recommend “UE4 Fundamentals” guide**.

In this section of the guide you will learn:

- How to download and install UE4
- Starting your very first project
- Launching the editor
- Setting preferences for the first time after launching the editor

- Creating, opening and saving maps
- Working with viewports
- How to navigate around viewports
- How to move/rotate/scale objects
- How to work with Local/World coordinate system
- Adjust details properties for each object
- Using the Content Browser to insert Static Meshes into your map
- Using grid snapping
- Adding and subtracting with BSP brushes
- Player scale
- Inserting a Player Start
- Inserting sun light
- Inserting point lights and spot lights
- Using the World Outliner
- Play testing your level from the editor
- Building (rendering) your level

Let's begin.

---

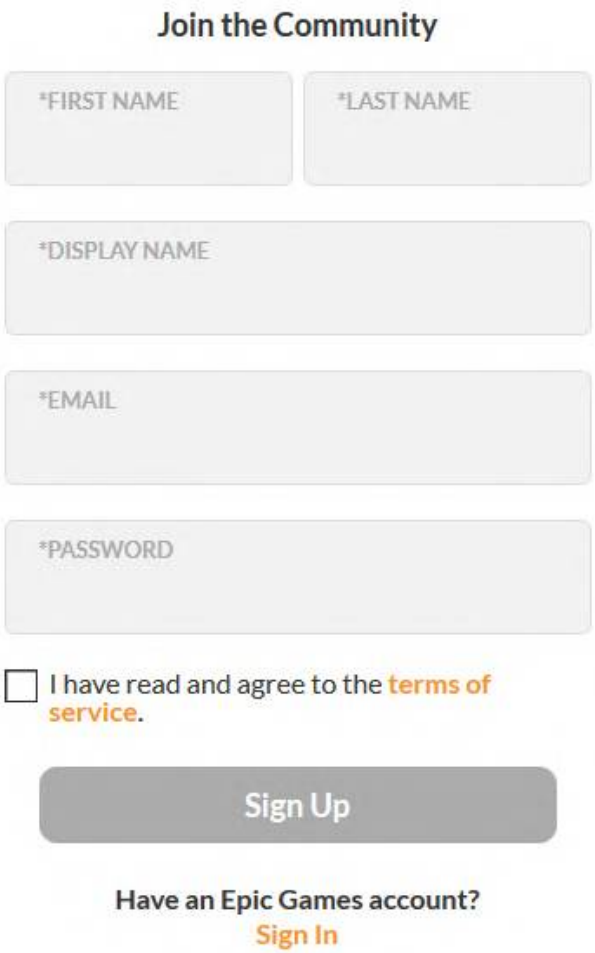
1. DOWNLOAD AND INSTALL

Unreal Engine 4 is free to use.

Let's download and install it.

Go to https://www.unrealengine.com/ and click on "Get Unreal" icon on the top right of the website:

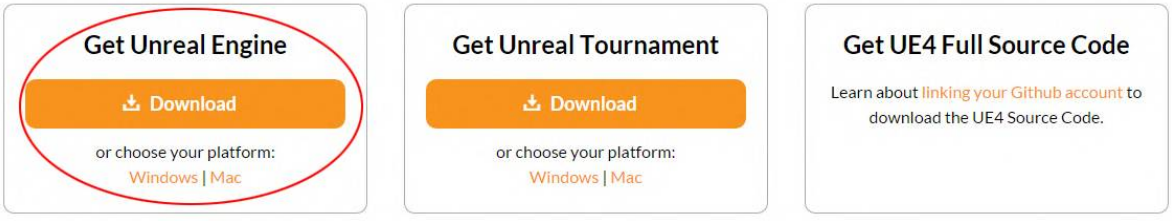
You will need to create a log-in and register to access Unreal Engine 4:



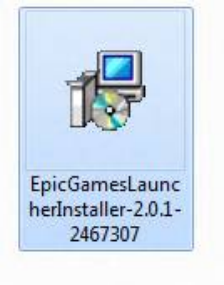
Before you can use the game engine, you have to download and install Unreal Engine/Epic Games Launcher.

Unreal Engine Launcher is very important. It is a portal through which you download and launch any Unreal Engine version you want. Through this portal, you also create new projects, manage your projects and download Marketplace Content.

So go ahead download:

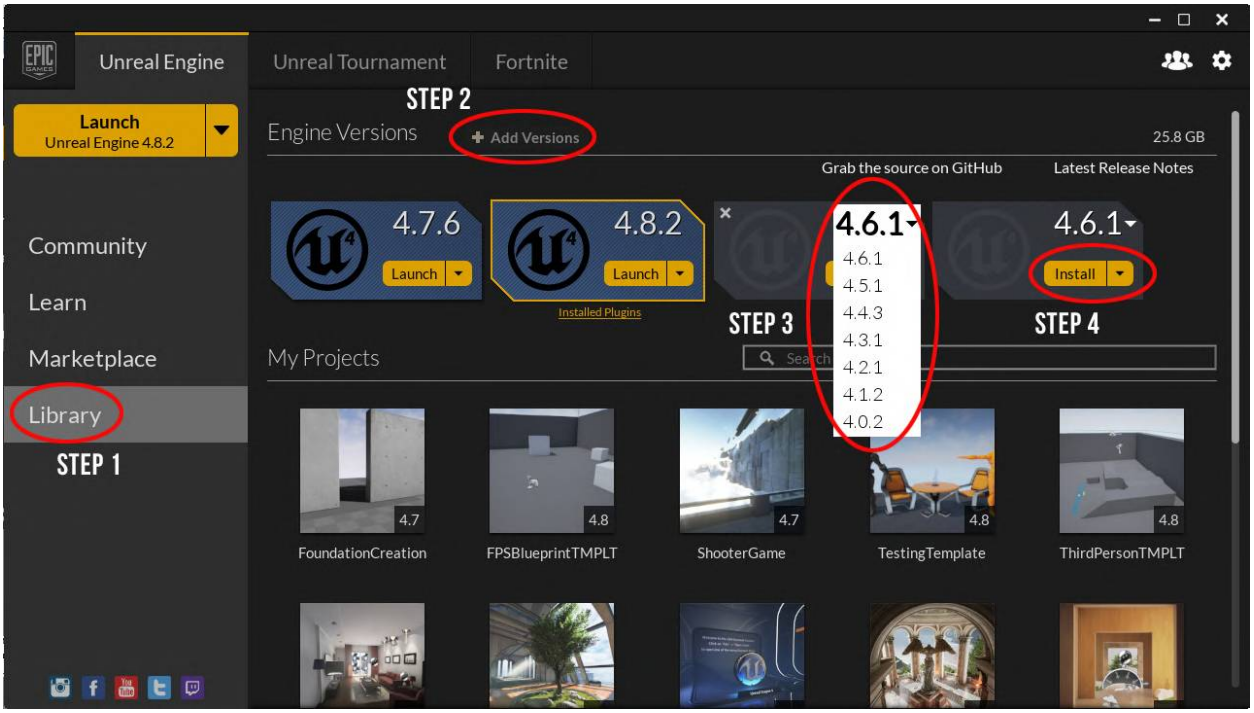


And then install “Epic Games Launcher” file:



After installation, you should now have “Unreal Engine 4” or “Epic Games Launcher” shortcut on your desktop, double click to open it:

Click over to Library tab and under Engine Version click on the plus icon. Using the drop down menu, choose which Unreal Engine version you want to download and install. Choose the newest/latest version to download:



Downloading and installation will take a bit of time, but once it is done, you now will have Unreal Engine 4 on your computer - ready to use.

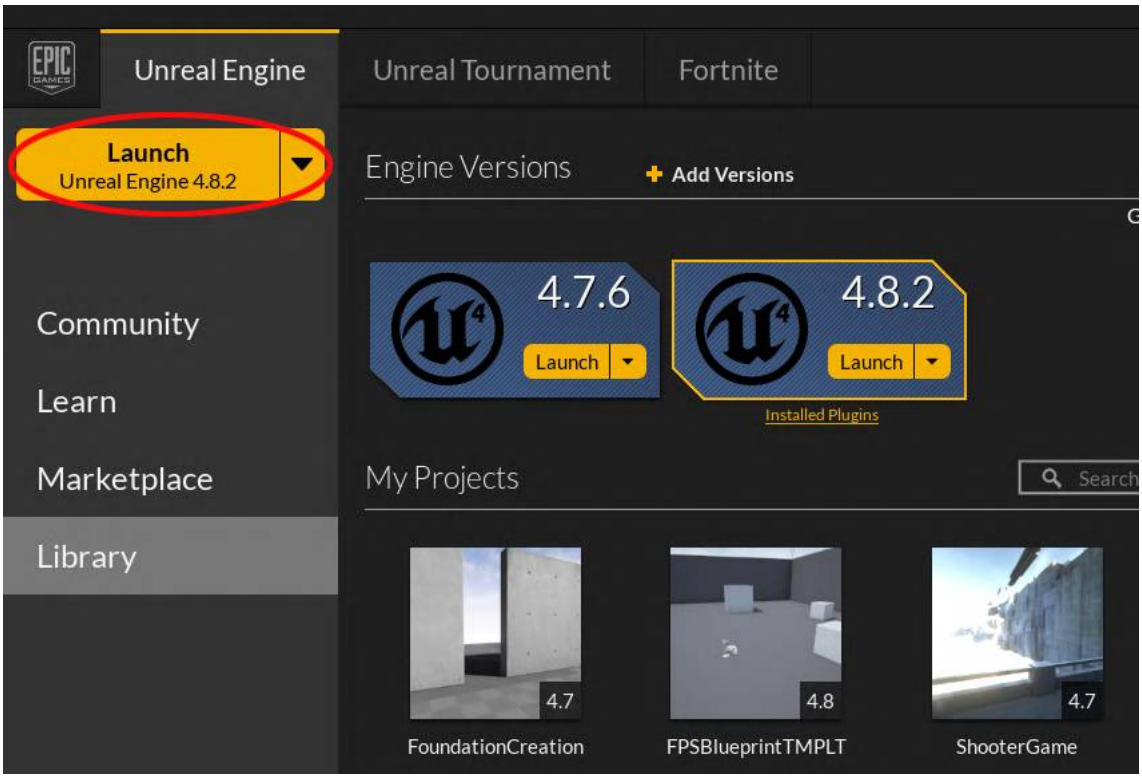
2. CREATE YOU FIRST PROJECT AND LAUNCH THE EDITOR

In order to start Unreal Engine 4 editor you will have to:

- Create a new project
- Open already existing project
- Open a project that you downloaded from Learn/Marketplace section

Let's create our first project.

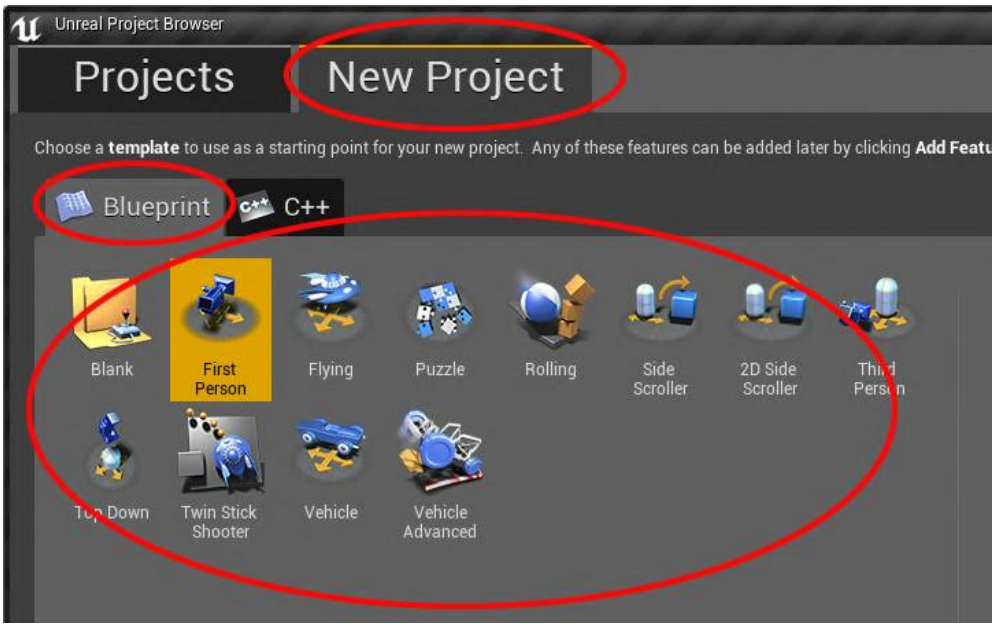
In "Unreal Engine/Epic Games Launcher" left click on "Launch Unreal Engine version" button on top left:



Switch tab to "New Project" and Choose Blueprint instead of C++.

Blueprint is a very powerful visual scripting language and you won't have deal with any C++ code.

Choose a game template to use. You can choose FPS, third-person or any other available templates:

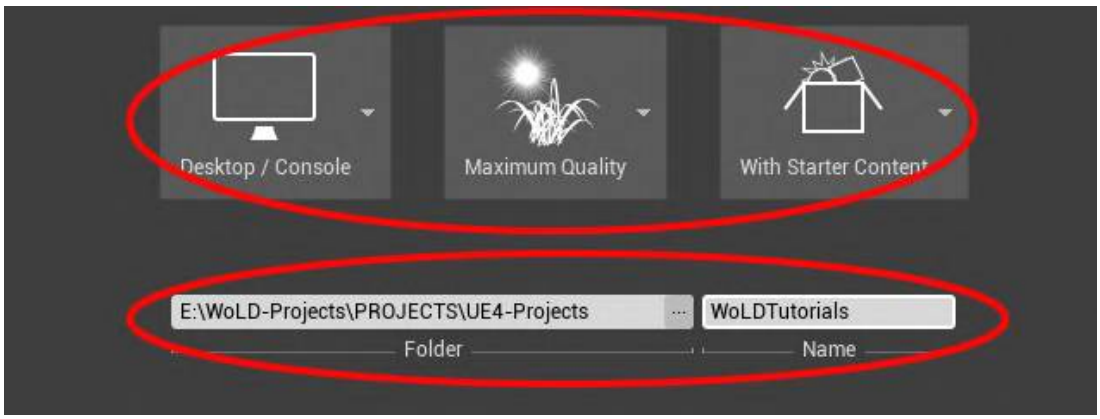


Once you get more advanced with UE4 you can learn C++ programming. But to keep things simple, begin with Blueprint. It will do most of the functionality you will ever need to start with - especially if you are a level designer or a game environment artist.

Next, choose the following:

- Desktop
- Maximum Quality
- With Starter Content.

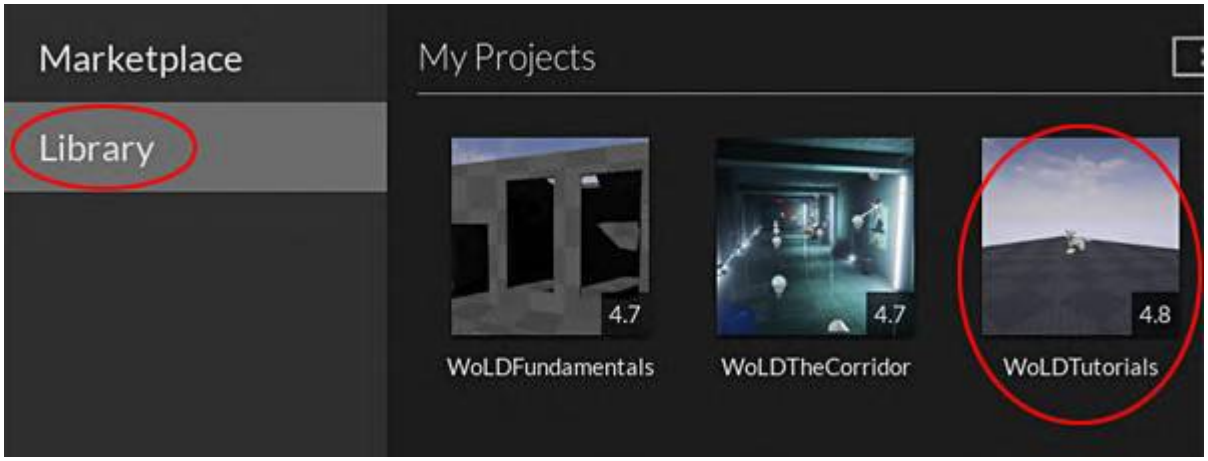
Select a location for your project to be stored. I choose to store all of my Unreal projects on a different drive other than my main C drive.



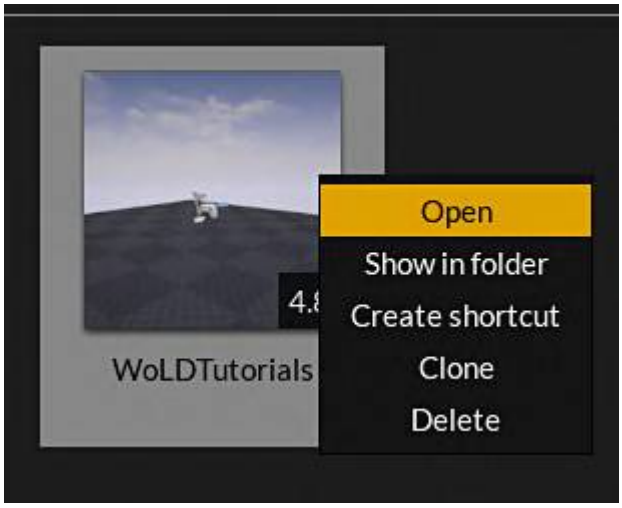
Name your project and click on Create Project:

This will launch the Unreal Engine editor and open your newly created project.

Once you created a project, you will see it show up in Unreal Engine Launcher under the Library tab:

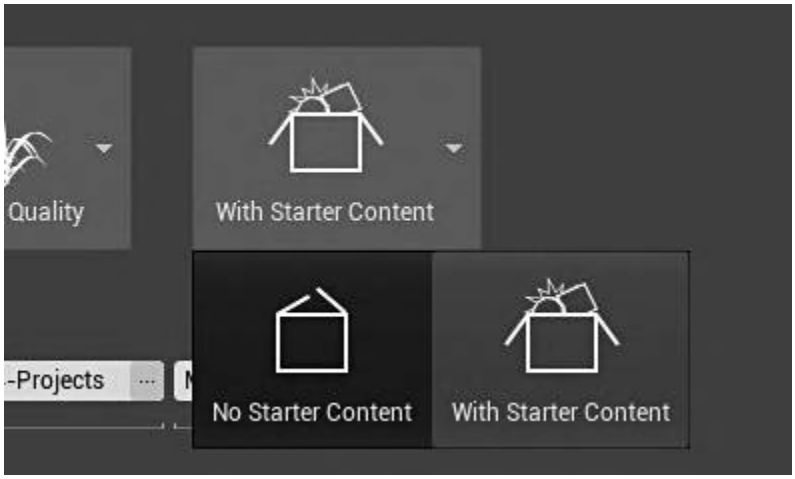


To open already created projects, double click on the icon within the Library tab or right click on the thumbnail and choose Open:

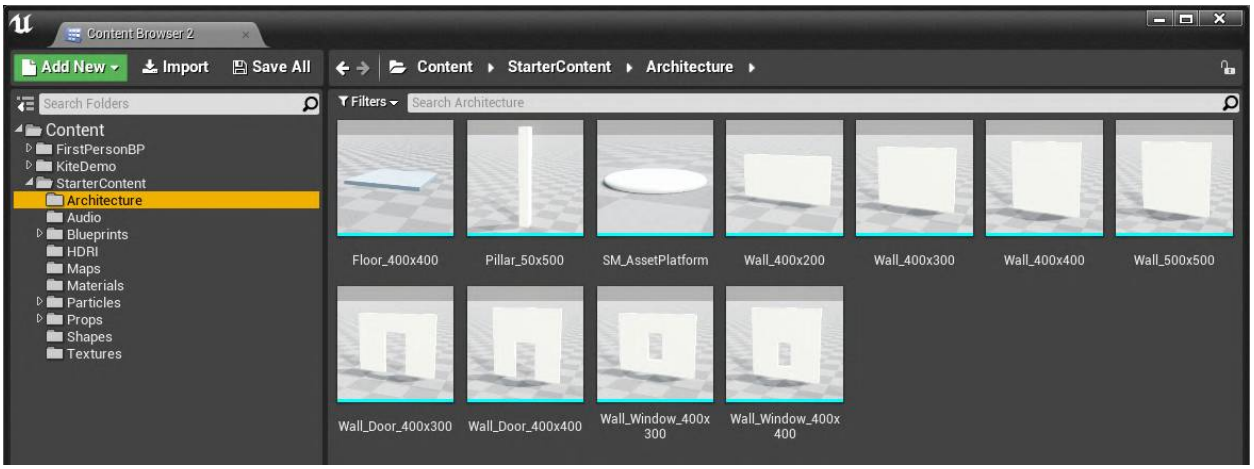


3. USE STARTER CONTENT

During project creation, make sure to enable "With Starter Content" option to have a set of UE4 assets to be available with your project:

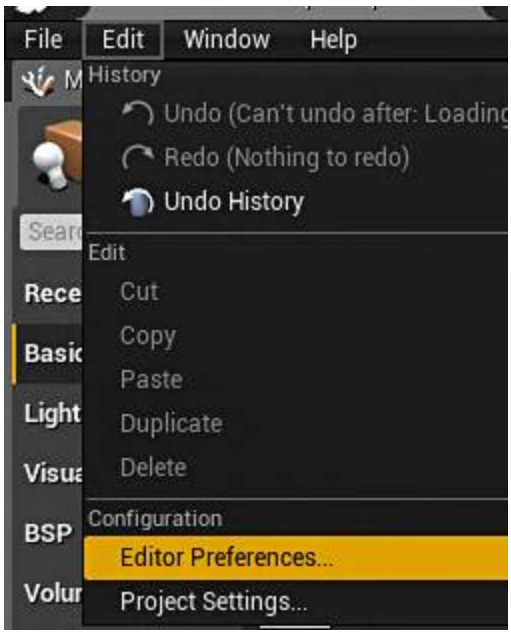


This will include a series of Static Meshes, materials and effects that you can use to construct a simple environment with:

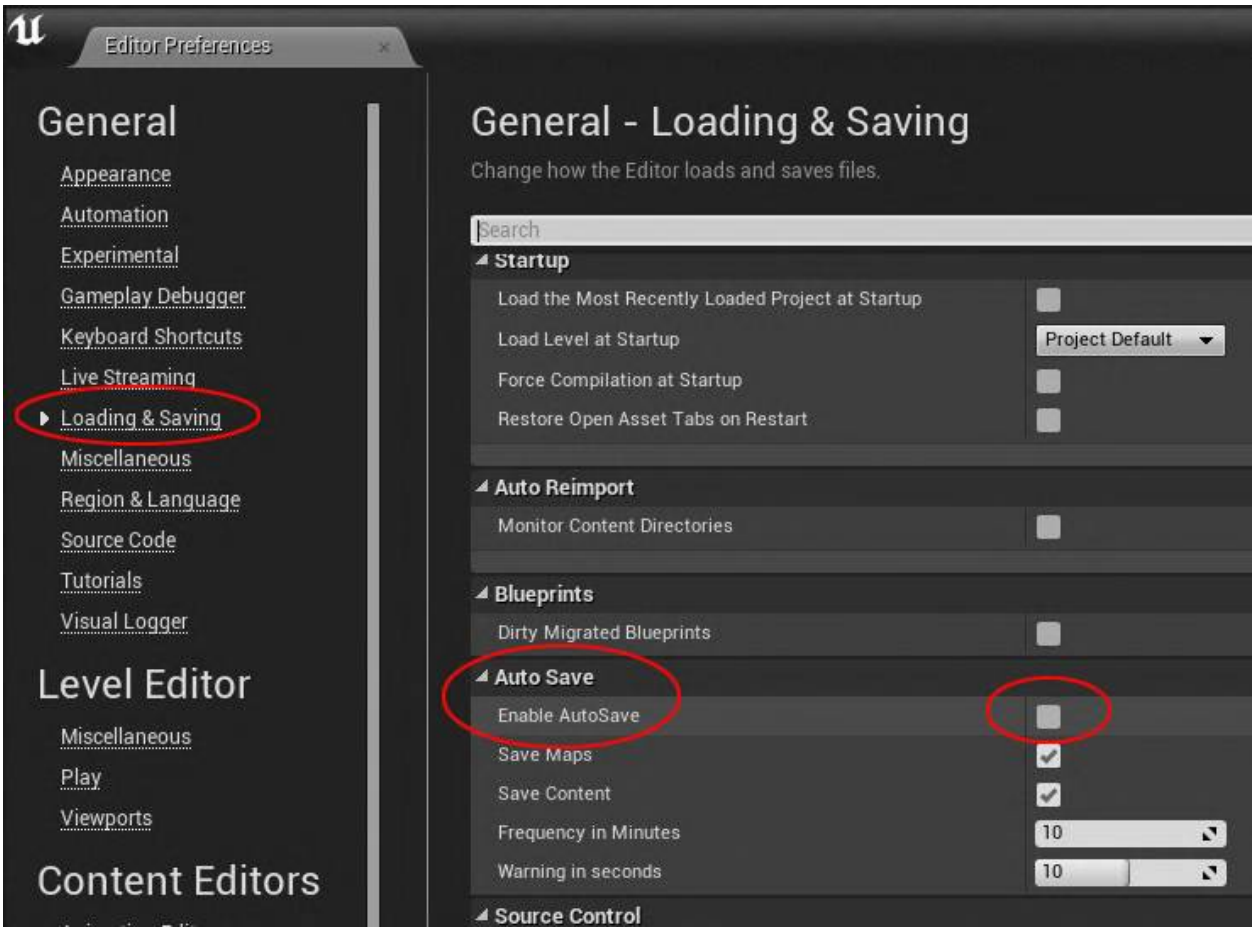


4. FIRST TIME INSIDE THE EDITOR

After launching the editor for the first time, go to Edit > Editor Preferences:

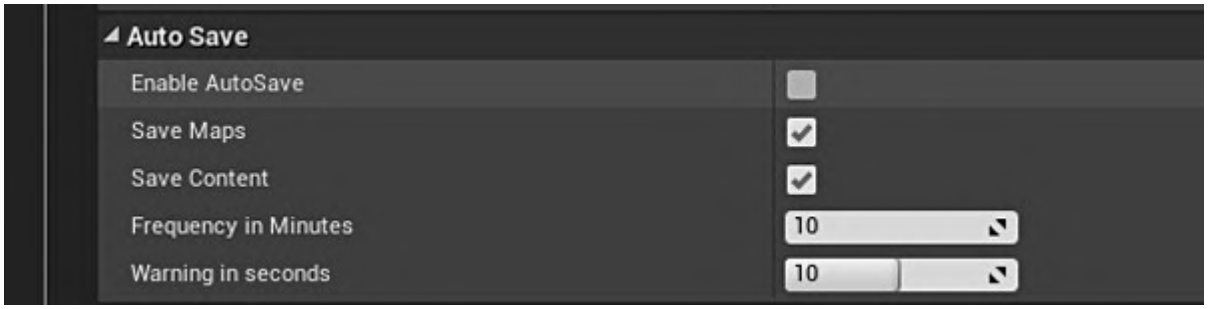


Under Loading & Saving and Auto-Save section, choose to Disable Auto-Save:



Now, this option is a personal preference. I disable auto-save functionality because I like to control when I save my work. But, this is a very useful function to keep enabled, if you want.

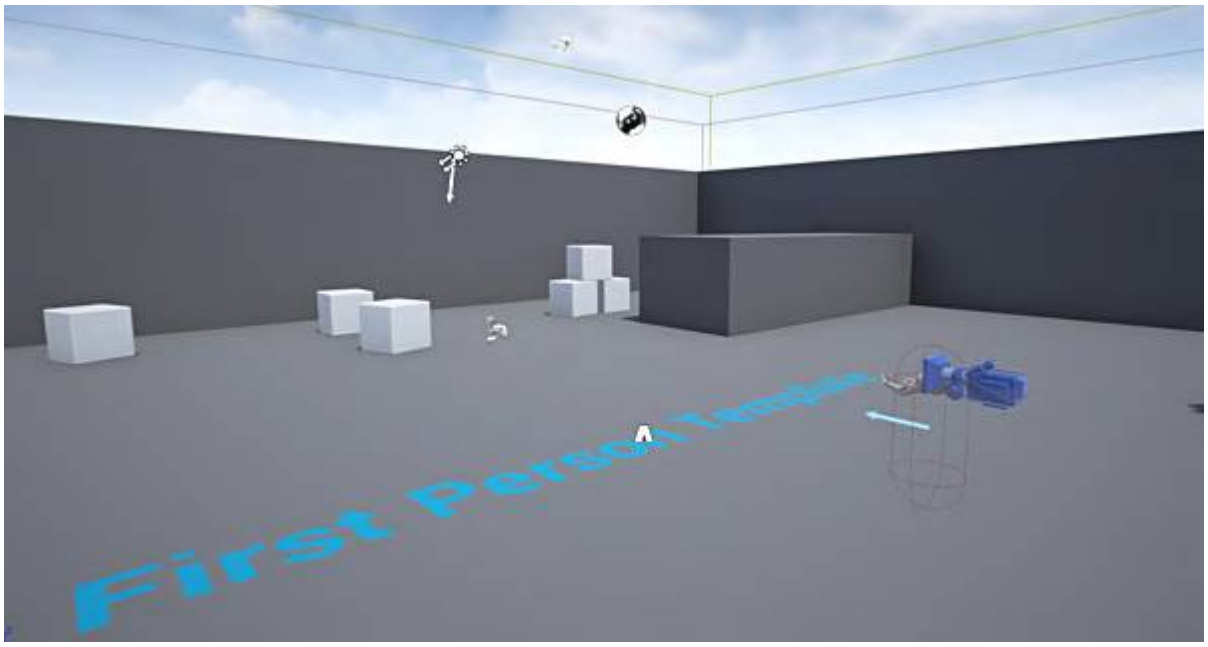
If you choose to keep Auto-Save enabled, set how often you want the editor to automatically save your work and how often to warn you about when it is saving:



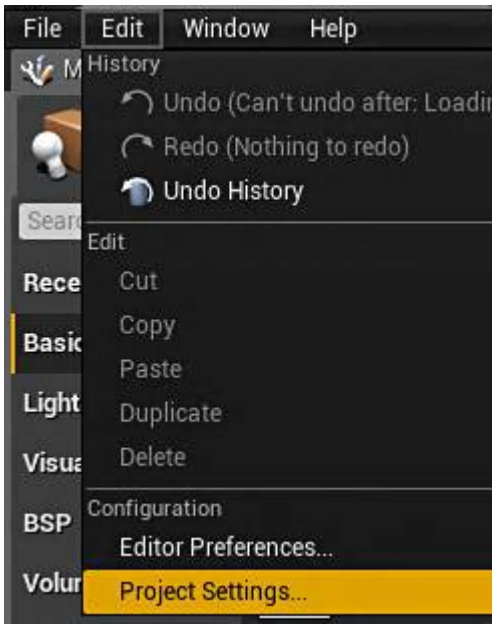
5. CREATING, SAVING AND OPENING LEVELS/MAPS

As in any game engine/level editor you can create, save and open maps. The file extension for UE4 maps is .umap.

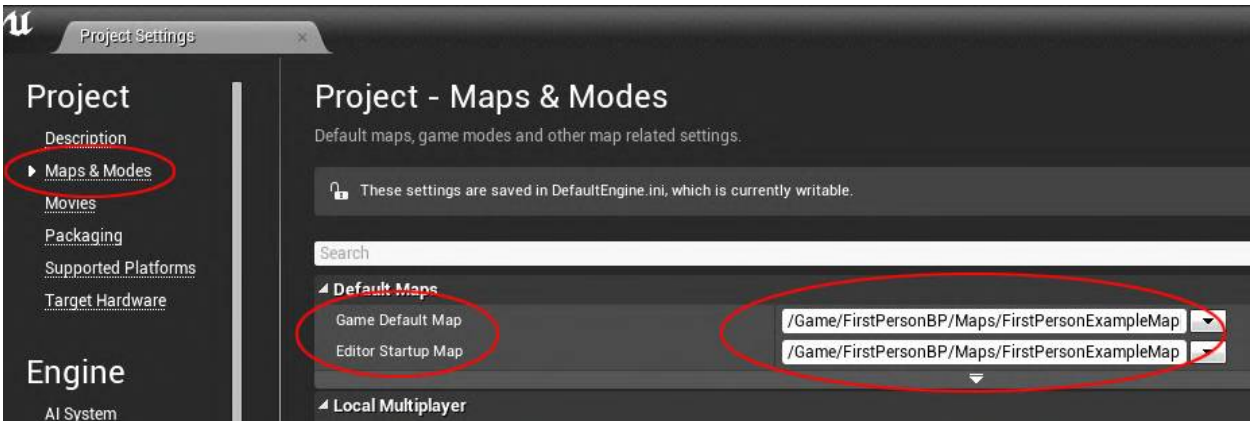
Default Maps: As you launch a project, you will have a default level/map open automatically. Depending on the type of game template you chose, this starter map will vary. Default starter map from FPS Shooter Template:



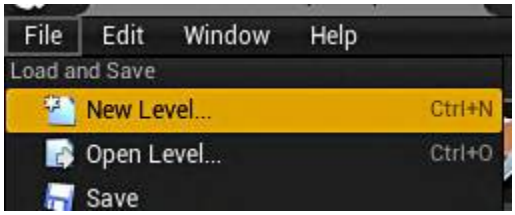
You can change starter default map to any other map. To update this go to Edit > Project Settings:



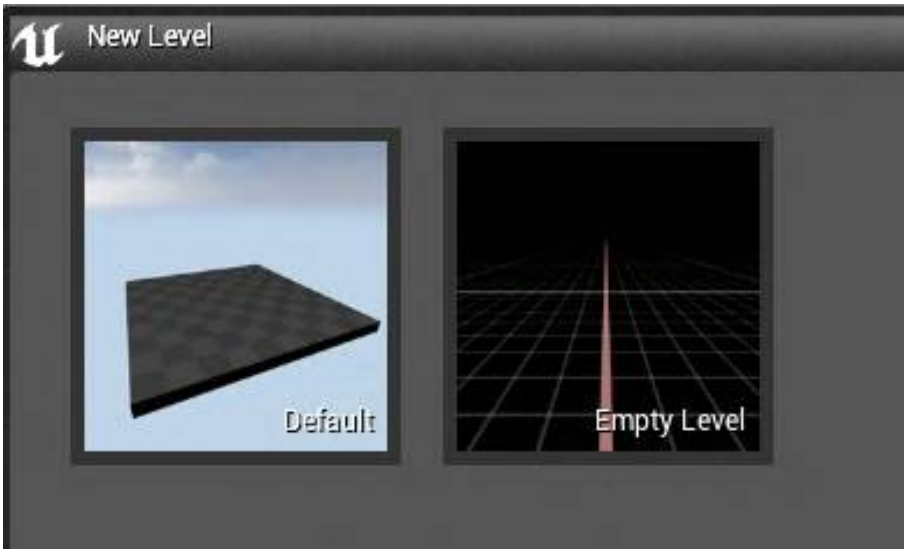
Under Maps and Modes and Default Maps you will have an option to switch to Game Default Map and Editor Startup Map:



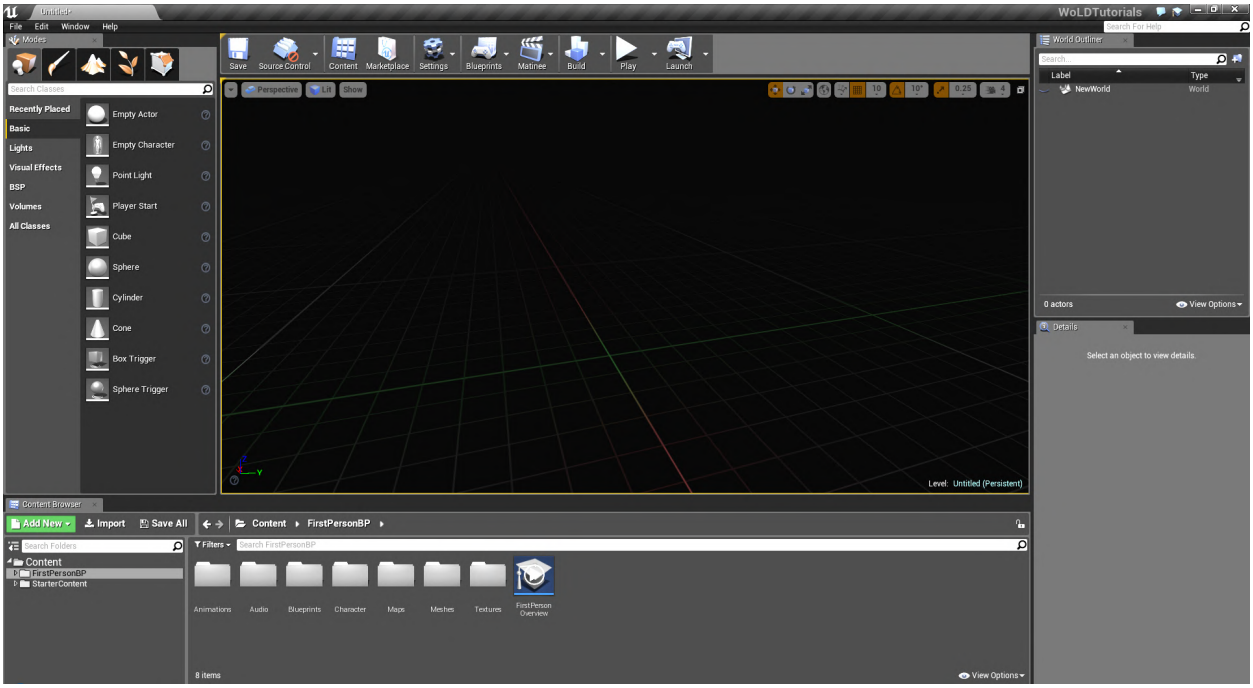
Start a New Map: To start a new blank map without any actors go to File > New Level:



Choose between Default and Empty Level:

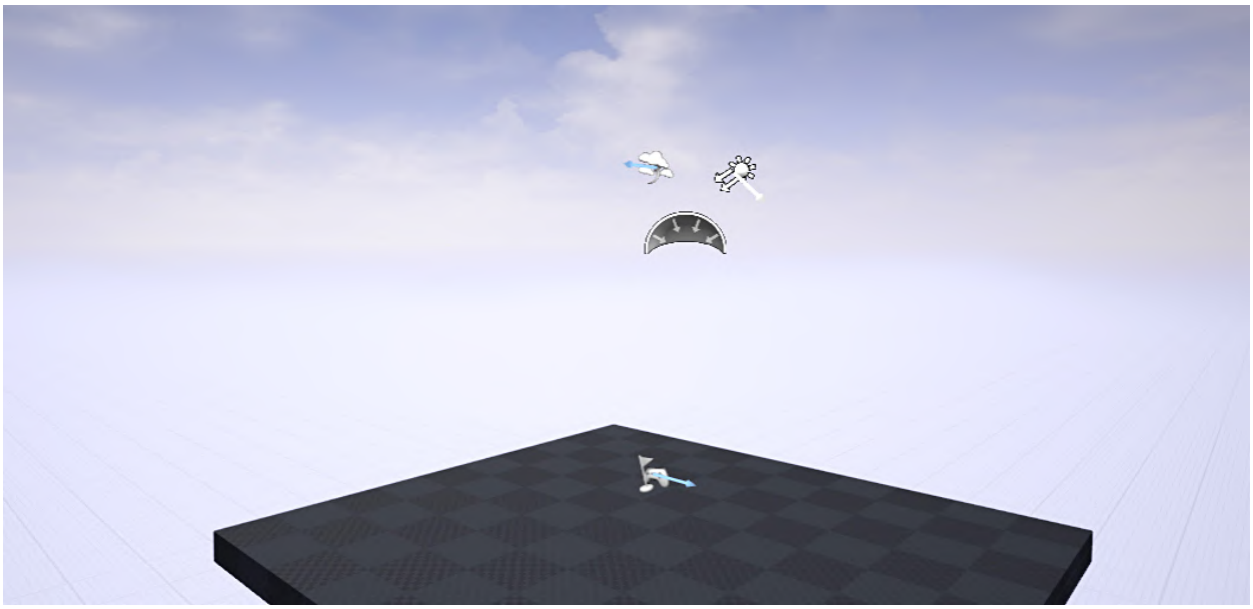


Empty Level will be a blank map without anything in it:

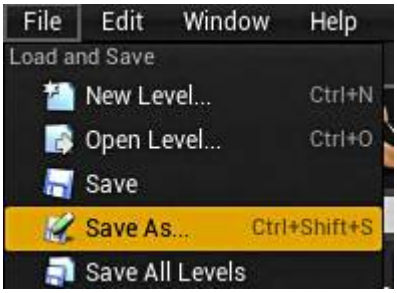


But to get things started quicker, choose a Default Level.

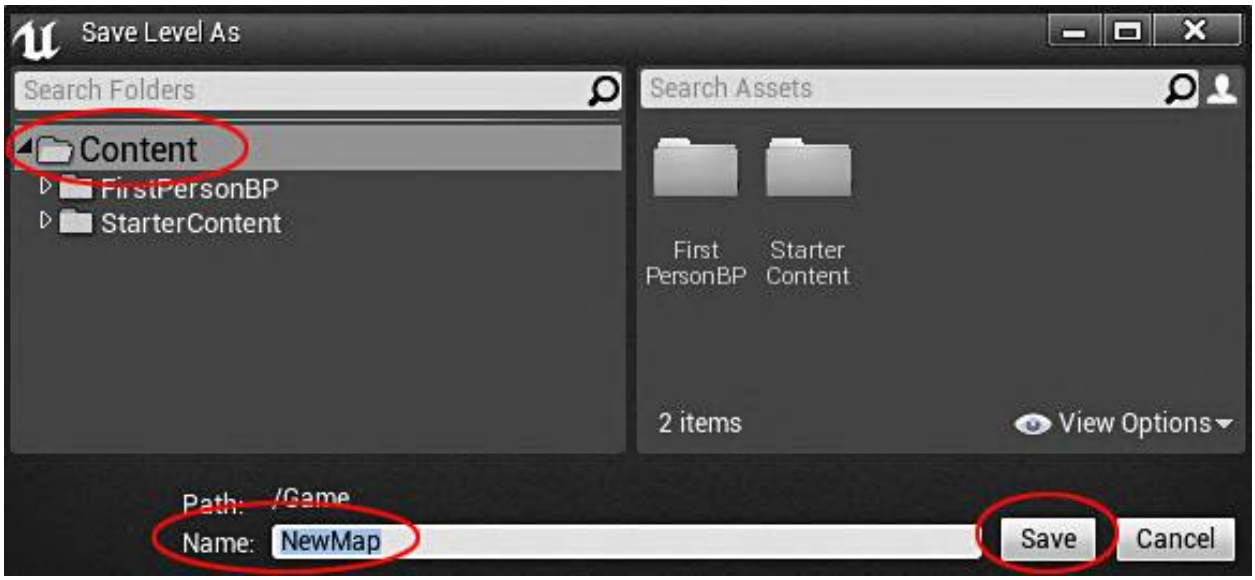
Default template gives you few necessary actors to start with such as a ground plane, light source (Directional Light and Skylight), player start, skysphere and atmospheric fog:



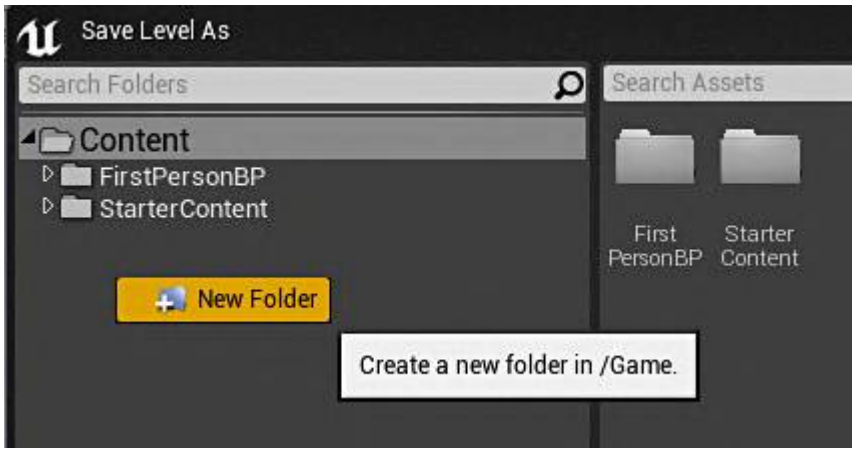
Saving Levels: To save any level you are currently working on, go to File > Save As:

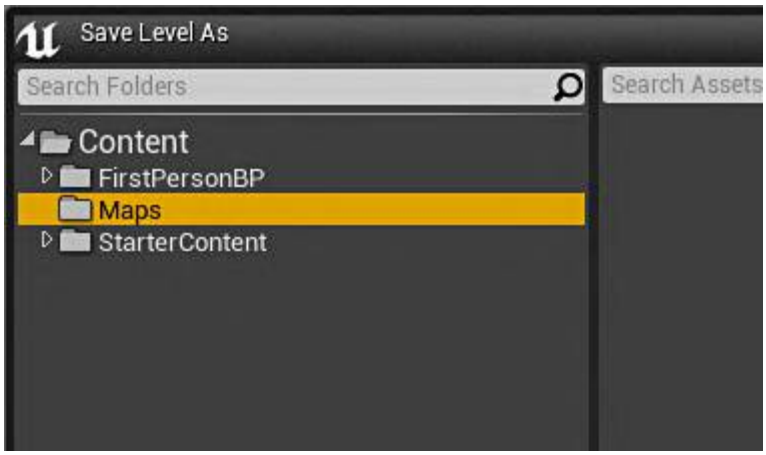


Choose a folder to store your map in, name it and choose Save:

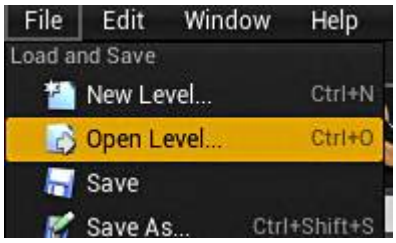


Make sure that all of your maps and any content that you save are inside the Content folder of your project**.** Do not save anything outside the Content folder. You can organize your maps better by creating a new folder inside the Content folder and naming it Maps:

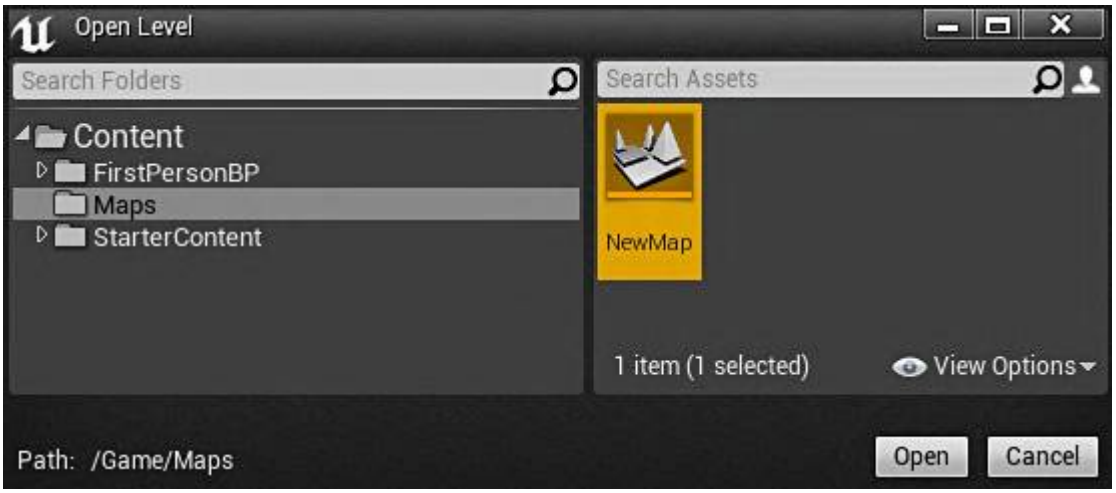




Opening Levels: To open any existing/saved level, go to File > Open Level:



Navigate into the folder where the map is saved and click Open:



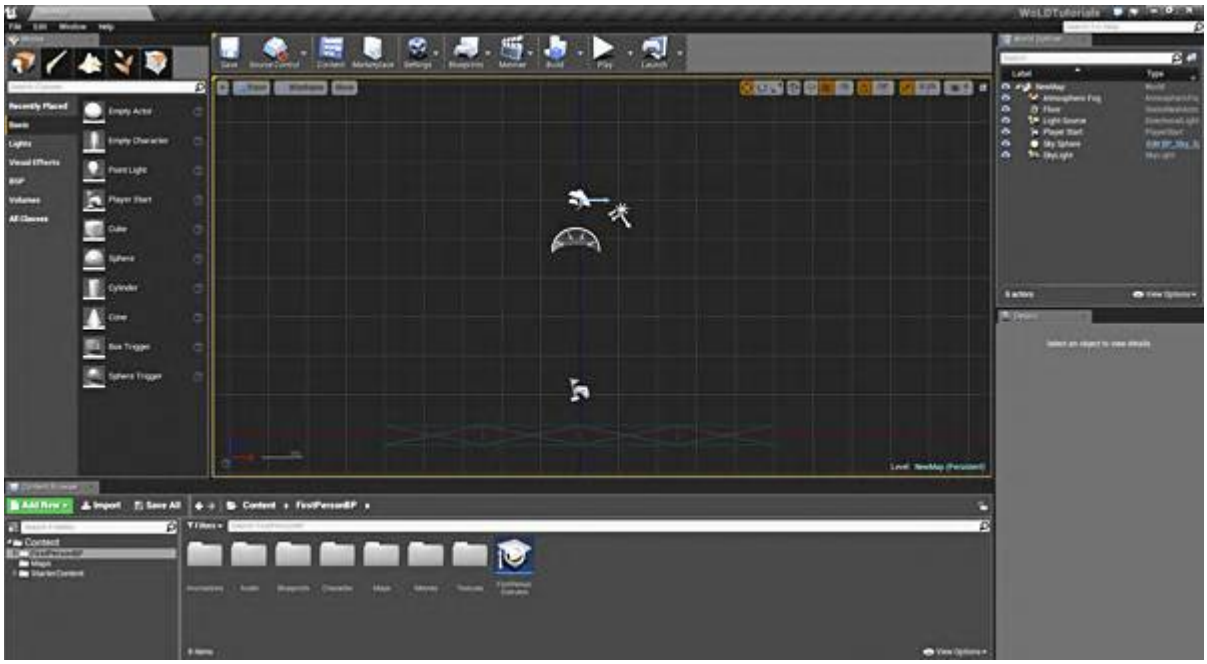
6. HOW TO WORK WITH VIEWPORTS

There are two types of viewports: perspective and orthographic.

Perspective is the real world view. It is how your level looks from the point of view of the player inside the game:

You could also enable perspective viewport as a wireframe, unlit and other various other developer modes.

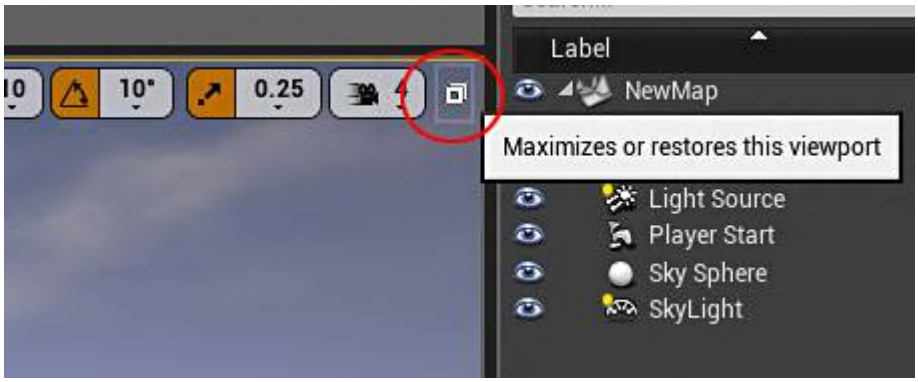
Orthographic view is the schematic wireframe grid view. You will have a front, side and top views:

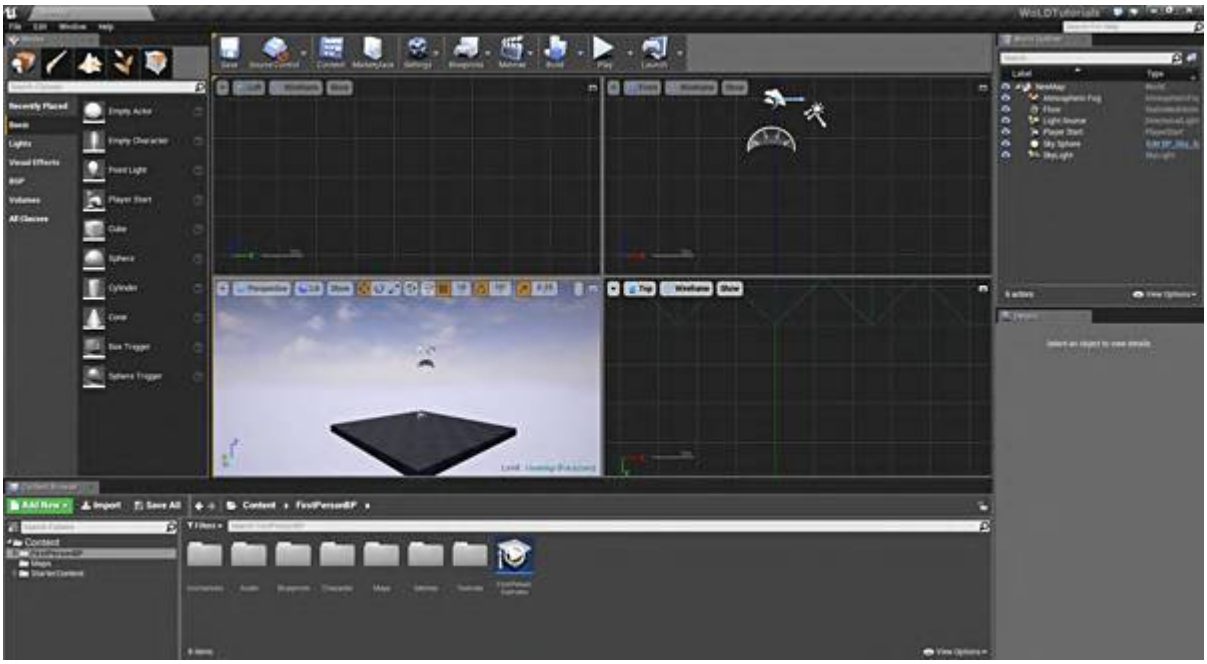


Perspective is going to be your primary view through which you work and construct your map.

Switch to 2x2 view of the editor by restoring the perspective viewport.

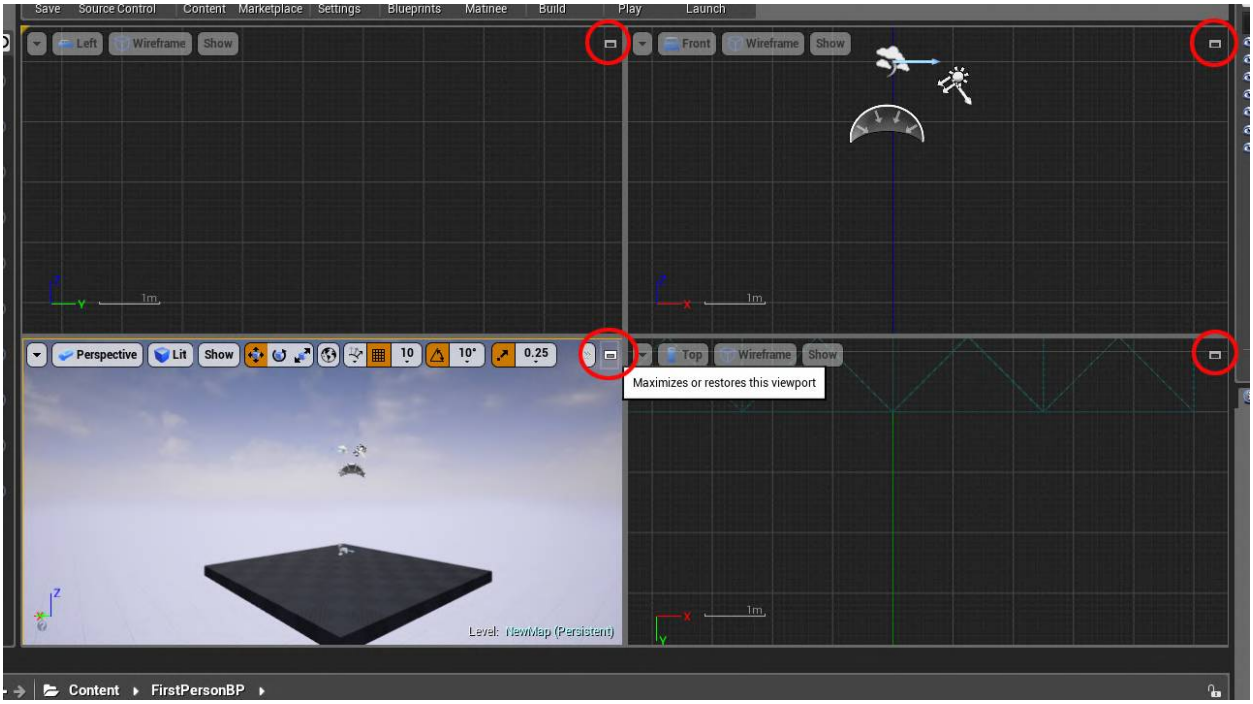
Click on "Maximize/Restore this viewport" icon:





You will now have all 4 views on the screen at the same time.

Maximize any viewport to full window by clicking on the "Maximize/Restore this viewport" icon again:



Use the following hot keys to quickly switch to any orthographic viewport:

- Alt + G = Perspective
- Alt + H = Front
- Alt + J = Top
- Alt + K = Left/Side
- Alt + Shift + K = Right/Side

Let's switch back to perspective view.

Game mode will show you how the environment will look during gameplay. All editor actors become hidden. Press G for Game Mode.

- G = Game Mode

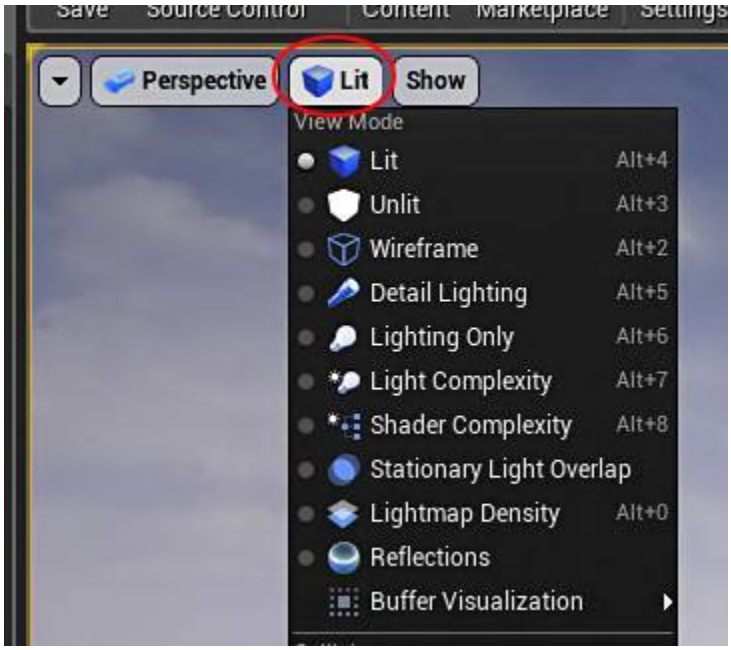
Real time mode will display animated and real time effects such as materials and particles. Press Ctrl + R for real-time mode.

- Ctrl + R = Real Time Mode

Pressing F11 will maximize the active viewport to fill the entire screen.

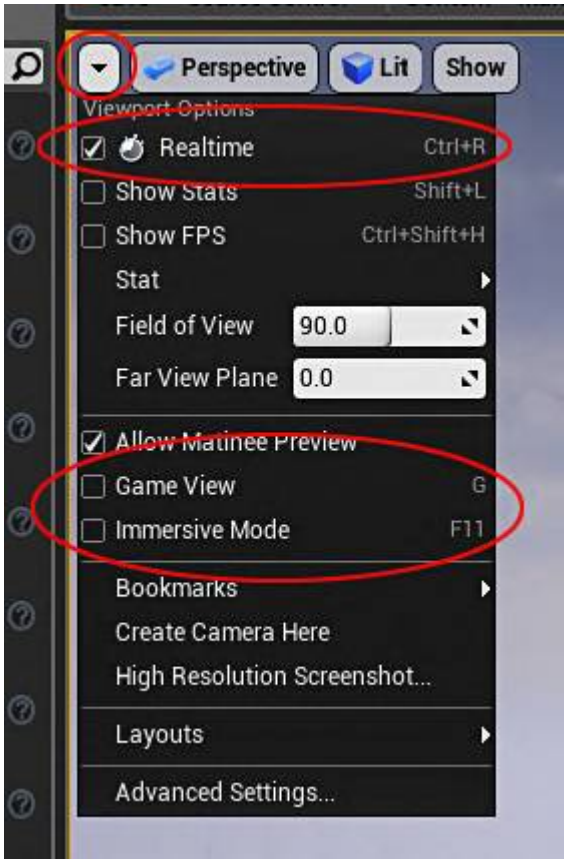
- F11 = Full View

You can change the perspective viewport to use different view modes, such as Unlit, Wireframe, Detail Lighting etc. Use the drop down menu to choose:



Each view mode has its own hot key. For beginners, keep your perspective viewport to Lit (Alt+4).

There are other handful of options under Show and Arrow drop down menu; including Realtime, Game Mode and Full View/Immersive Mode:



But the ones we just covered are enough to get you started.

7. VIEWPORT NAVIGATION

Let's cover the essentials for navigating each viewport.

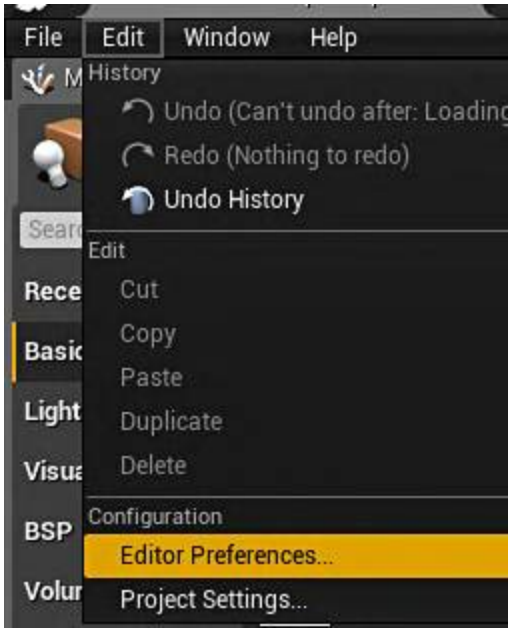
Few hot key abbreviations we'll be using:

- RMB = Right Mouse Button
- LMB = Left Mouse Button
- MMB = Middle Mouse Button

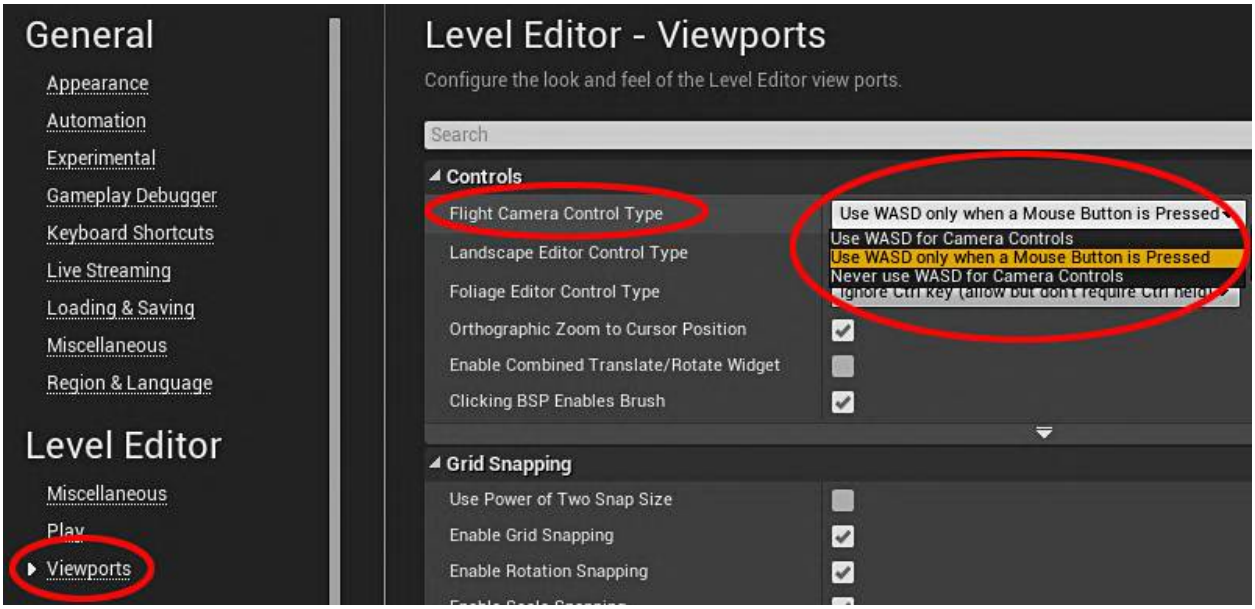
Navigating In Perspective Viewport: The main navigation key combination you will use the most is:

- Hold RMB + WASD = Move Forward/Back/Side-to-Side

To disable holding the Right Mouse Button as you press WASD keys to navigate, go to Edit > Editor Preferences:



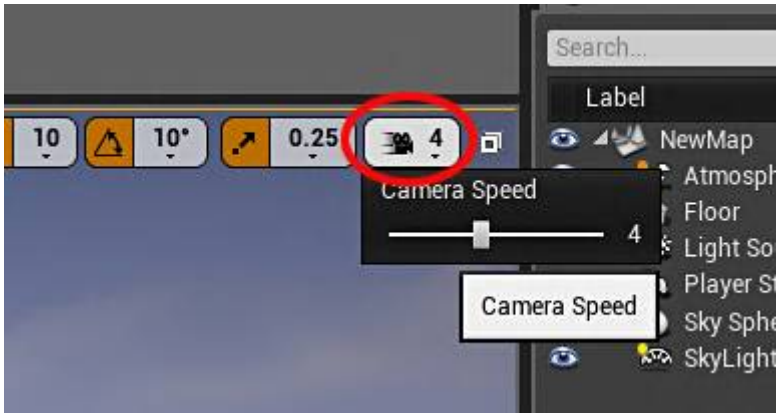
Under Viewports and Flight Camera Control Type, choose "Use WASD for Camera Controls" from drop down menu:



The rest of most common ways to navigate inside perspective viewport are:

- Hold RMB + Mouse Move Any Direction = Look Around From Stationary Position
- Hold LMB + Mouse Move Forward/Back and Left/Right = Move forward/back; Look left/right from stationary position
- Hold MMB + Mouse Move = Pan around Up/Down/Left/Right
- LMB+RMB+Mouse Move= Pan around Up/Down/Left/Right (same as MMB + Mouse Move)
- MMB Scroll = Camera zoom in/out

Perspective View Camera Speed: When you are navigating around in perspective viewport, the camera speed at which you look around can be adjusted. To change the viewport camera speed, use the slider to lower number (slower) or higher number (faster):



Navigating In Orthographic Viewport: Navigation in orthographic viewports is a lot simpler.

- Hold RMB + Mouse Move = Pan
- Hold LMB + Mouse Move = Marquee Selection
- Hold LMB + LMB and Mouse Move Forward/Back = Zoom In/Out
- MMB = Measuring Tool

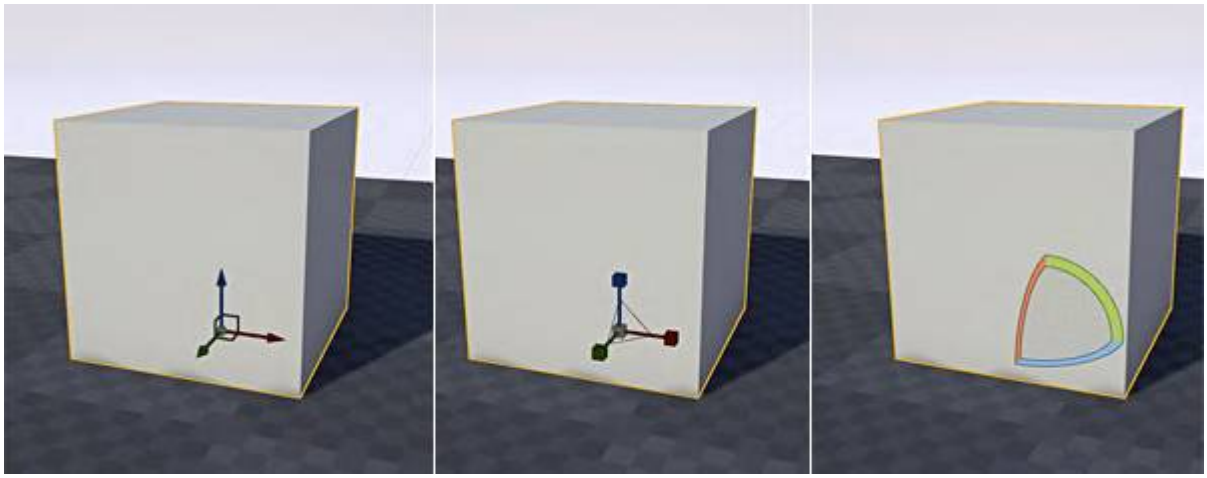
8. WORKING WITH ACTORS/OBJECTS (MOVE, ROTATE, SCALE)

In UE4, any object that you place in the world is called an actor. So we'll use actor and object interchangeably as they mean the same thing.

Very basic functions are selecting, deselecting and removing objects:

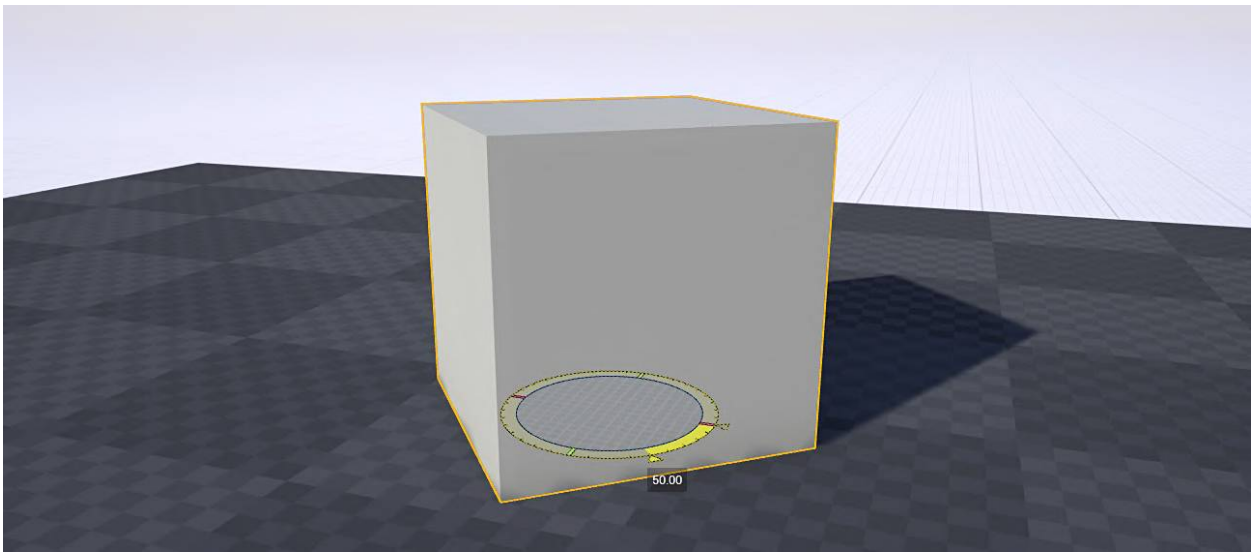
- Left Click = Select the Object
- Escape = Deselect the Object
- Delete = Remove the Object

After you select an object in the viewport you will see a move/rotate/scale gizmo appear:



- Spacebar = Cycle Between Move/Rotate/Scale Transformation Gizmo
- W = Move
- E = Rotate
- R = Scale

To move/rotate/scale the selected object, left-click and drag on the appropriate transformation handle within that gizmo:



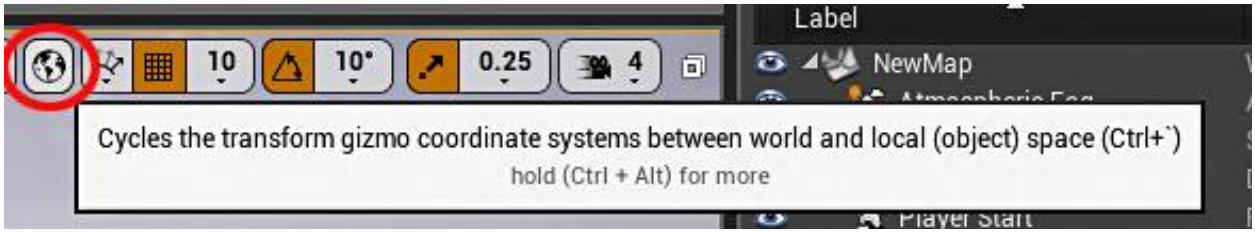
If you don't see the transformation gizmo, press G key to disable game mode.

9. LOCAL AND WORLD COORDINATE SYSTEM

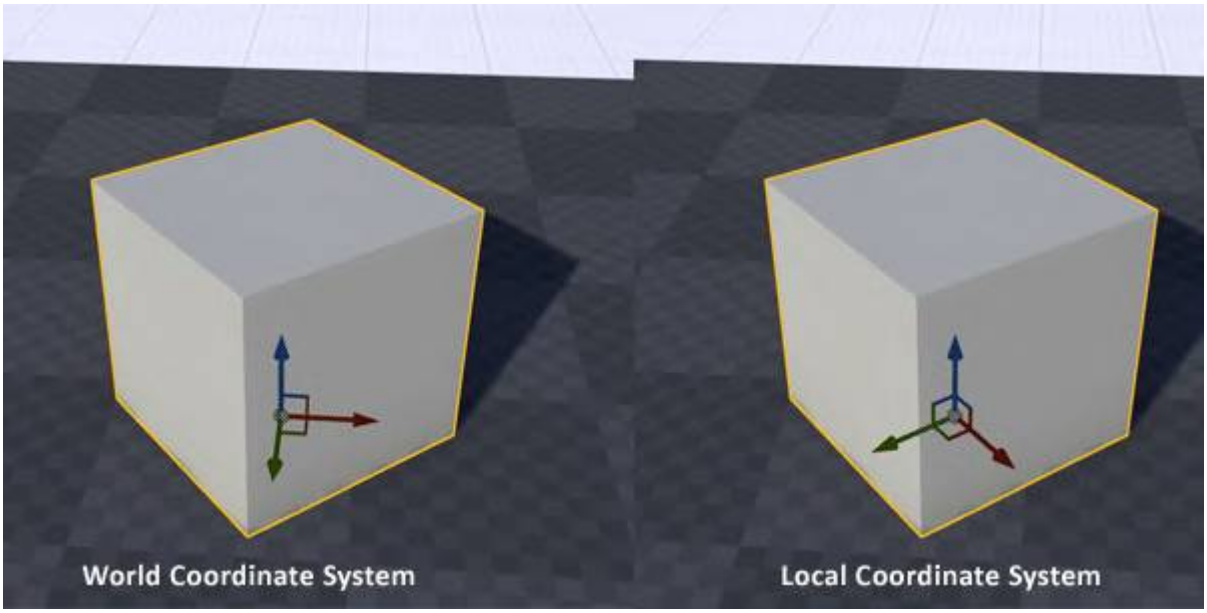
Each object in UE4 can be moved along 2 types of coordinate systems - world or local. Most of the time you will be working with World coordinate system as you construct your environment. But there will be time when you'll need to work with object's Local coordinate system.

- **World** coordinate applies to the entire world and XYZ direction does not change.
- **Local** coordinate applies to the individual object. It ignores the universal XYZ position of the world.

Click on the World/Local icon within the viewport to cycle World vs Local:



- Ctrl + ~ = Switch Between Local and World Coordinate System

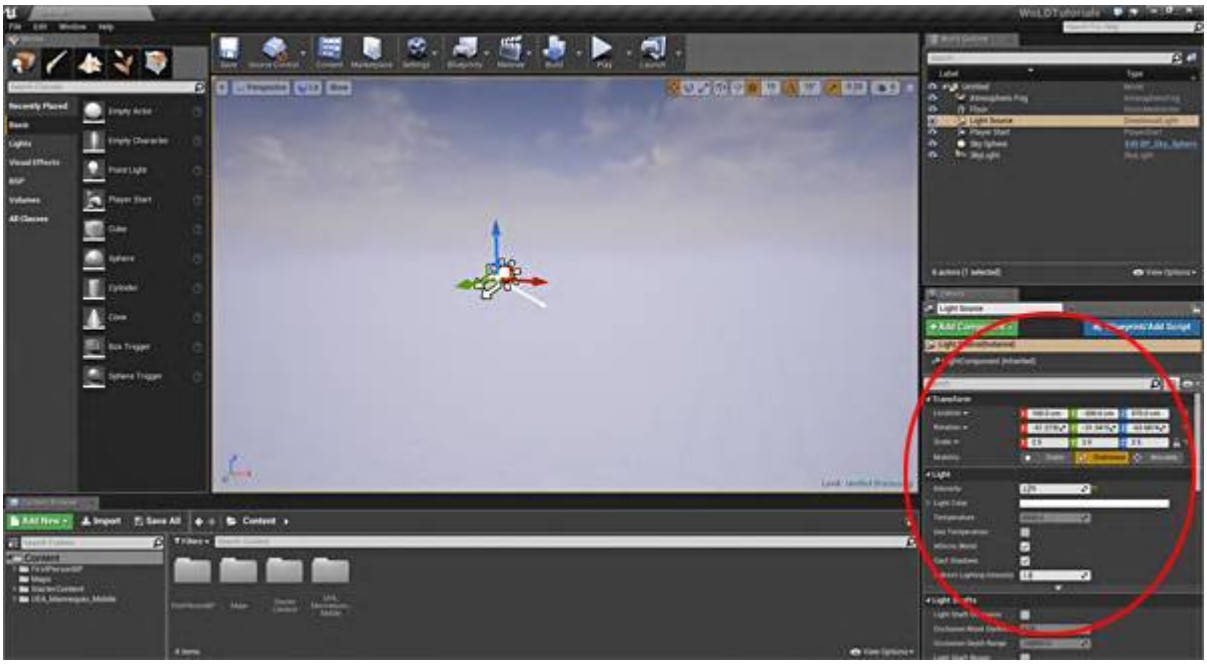


To see how this works follow these steps:

- Select any Static Mesh (3d model) inside the viewport
- Rotate it 40 or 50 degrees on Z axis
- Switch to Move tool (W)
- Move the box on X and on Y
- Switch from World to Local coordinate system (Ctrl + ~)
- Move the box again on X and on Y
- Notice the difference of how you are able to work along the World or Local coordinate system
- Switch back to World coordinate system (Ctrl + ~)

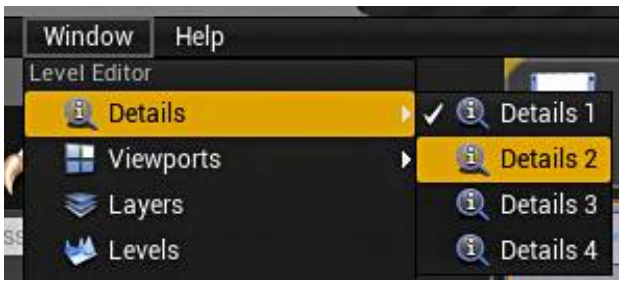
10. DETAILS PANEL / OBJECT PROPERTIES

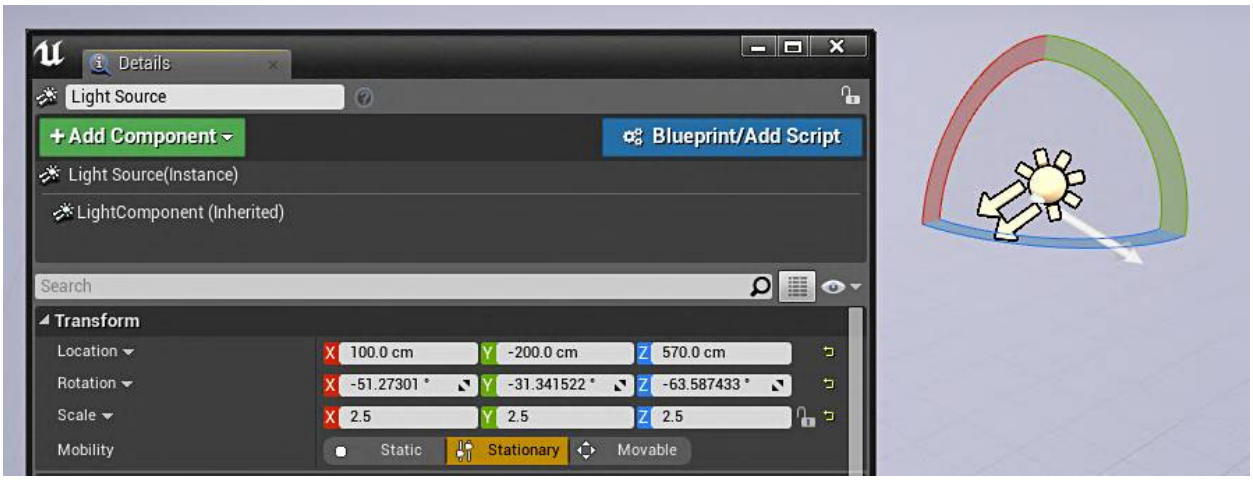
Each object/actor you select will have a set of properties to adjust. Select an object and take a look at the Details panel on the lower right hand side to see that actor's settings:



Depending on the type of actor you select, these settings will be different**.** For example if you select a Point Light, you can change color, light intensity and radius.

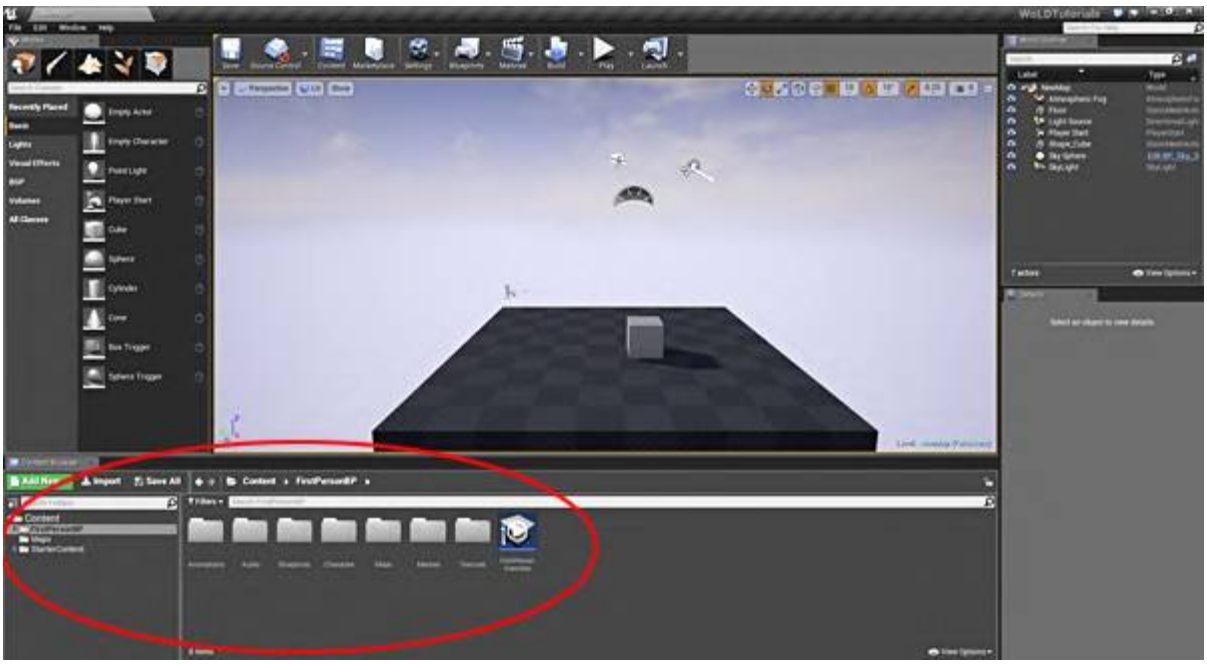
You can also open additional Details panels by going to Window > Details > Details 2, 3 or 4:





11. CONTENT BROWSER

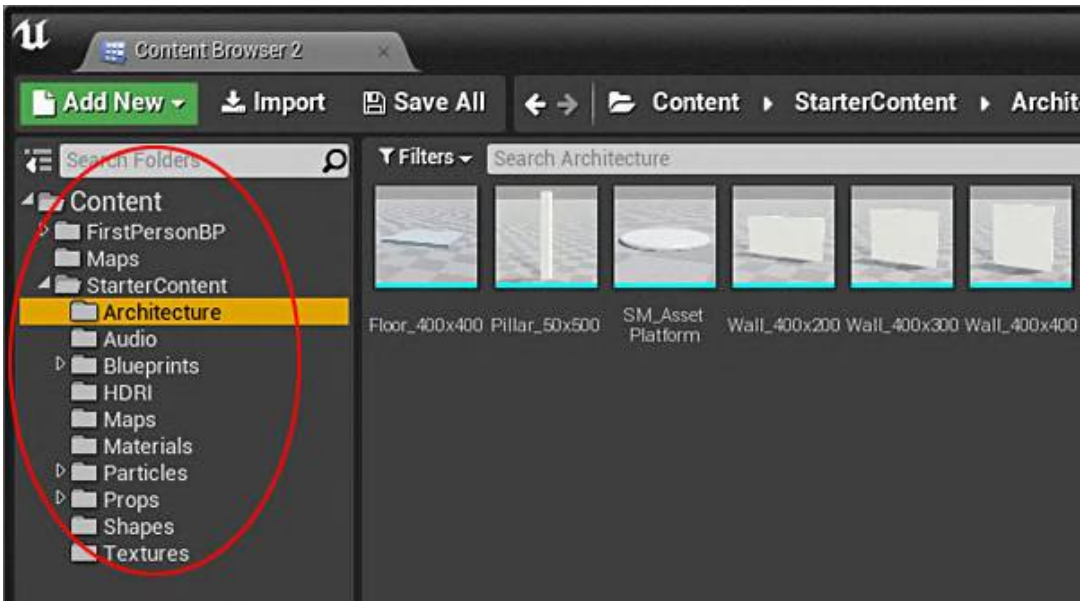
Content Browser is the content management system in UE4:



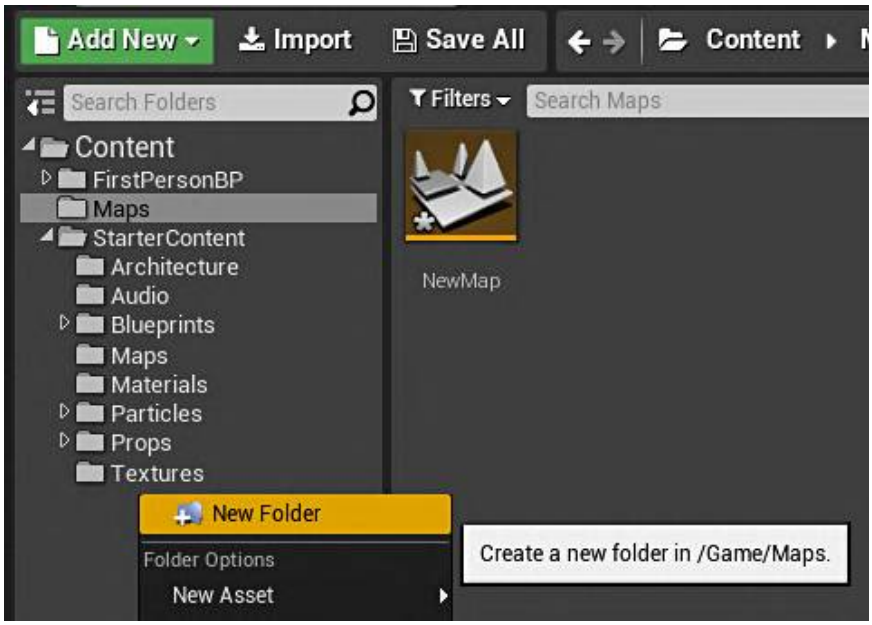
Through the Content Browser you will:

- Look for textures, materials, static meshes, animations, blueprints to insert into your level
- Import custom static meshes and textures
- Create new blueprints, materials, particle effects, etc

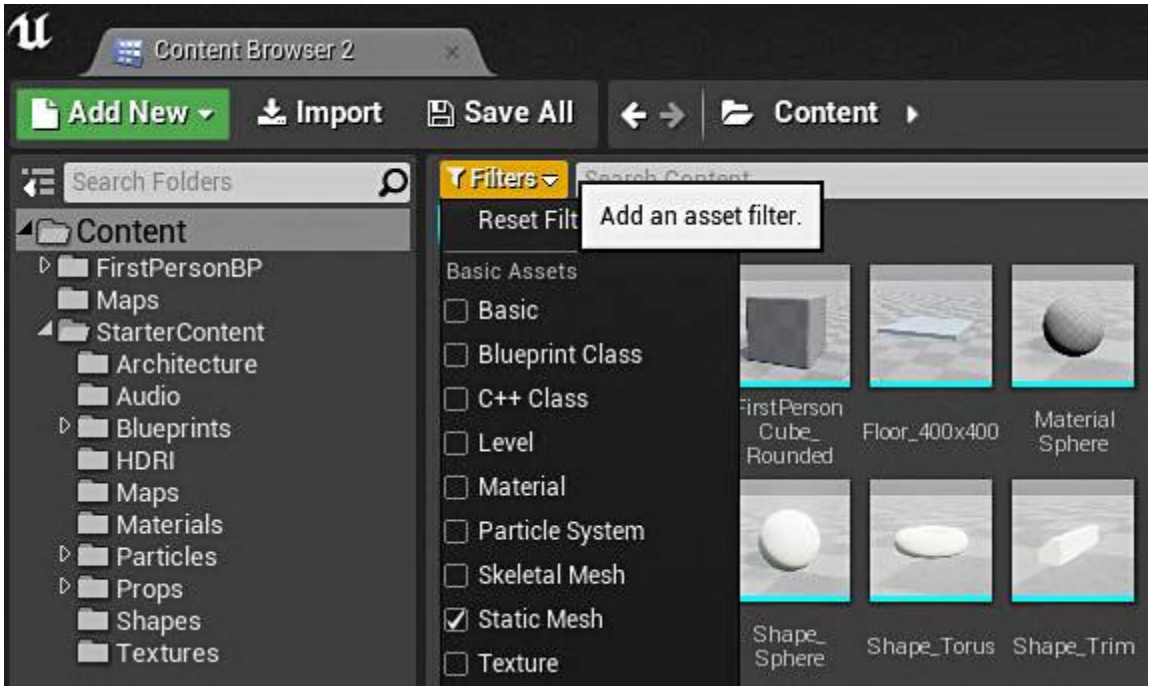
Use the folder structure to look into any folder and find the assets you are looking for:



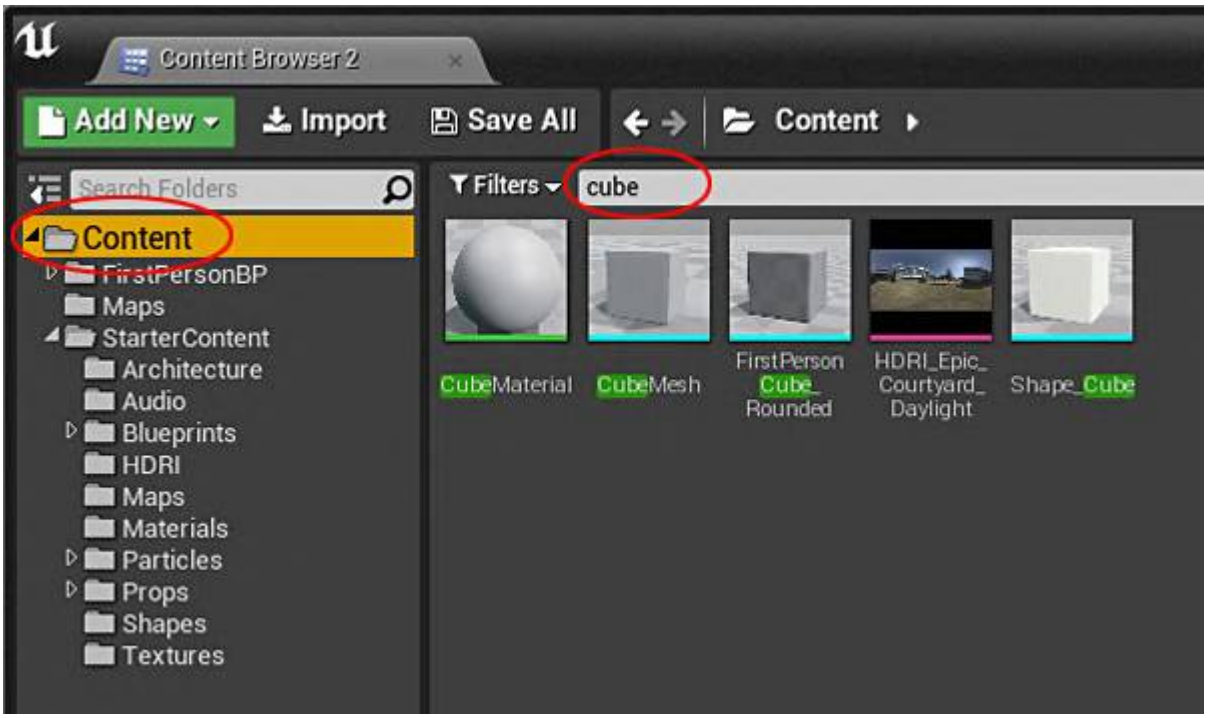
You can create new folders to better organize your assets:



Look for specific asset type by using the Filters drop down menu:



Use the search to look for a name of the asset you are looking for:



To search and to filter the entire directory of your project, make sure to select the Content Folder. This way you are searching everything within that folder.

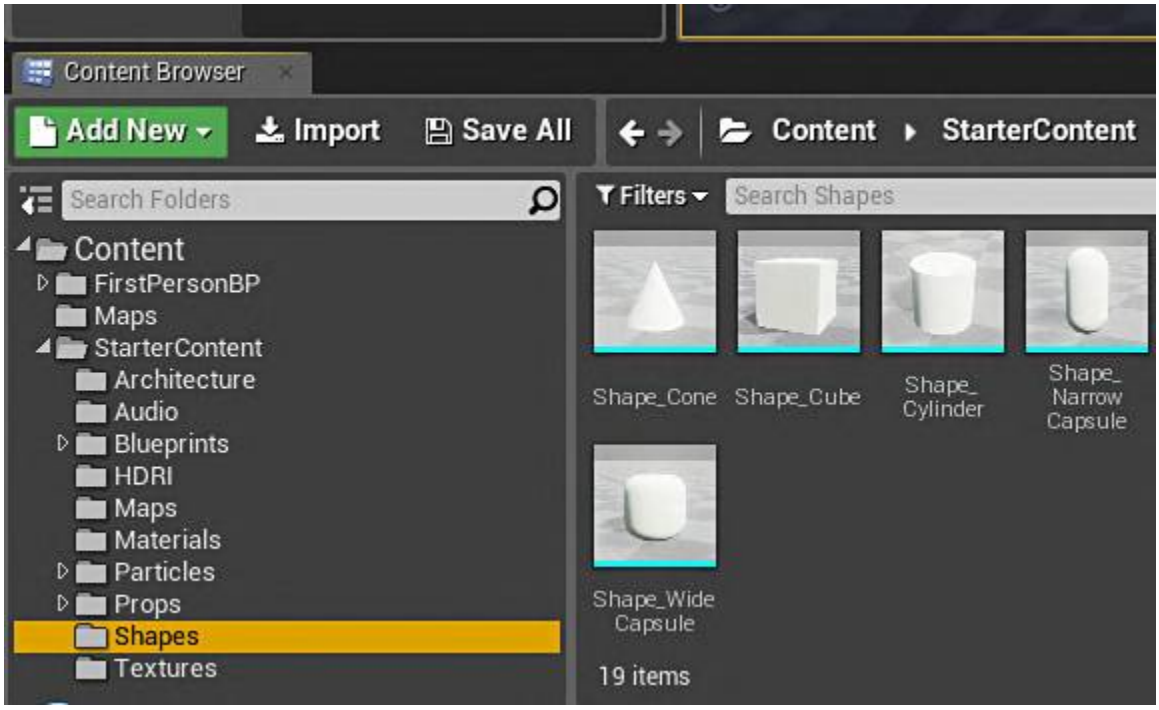
12. INSERTING STATIC MESHES

Static Meshes are 3d models and will make up the majority of your level construction.

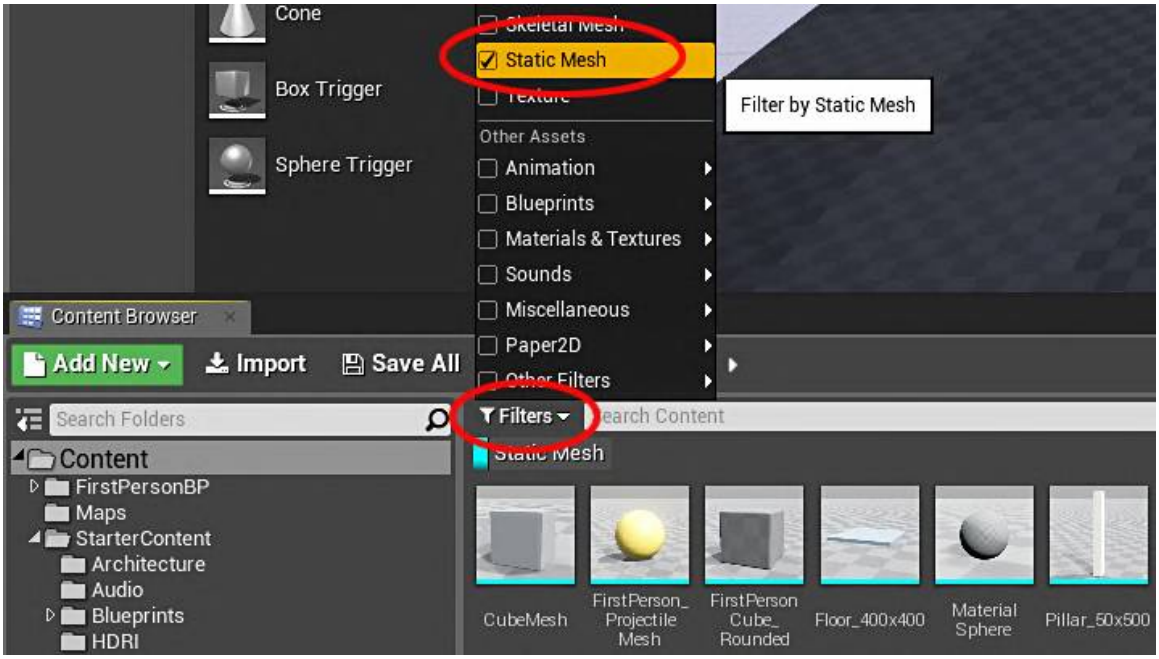
There are 3 ways you will have access to Static Meshes:

- You will use existing Static Meshes provided within a given project to construct the world.
- You could create your own custom Static Meshes using 3d modeling application such as Maya, Maya LT, 3ds Max, Modo or Blender.
- You'll have to find a set of Static Meshes to use created by someone other than yourself. You'll find some of them in Learn and Marketplace section.

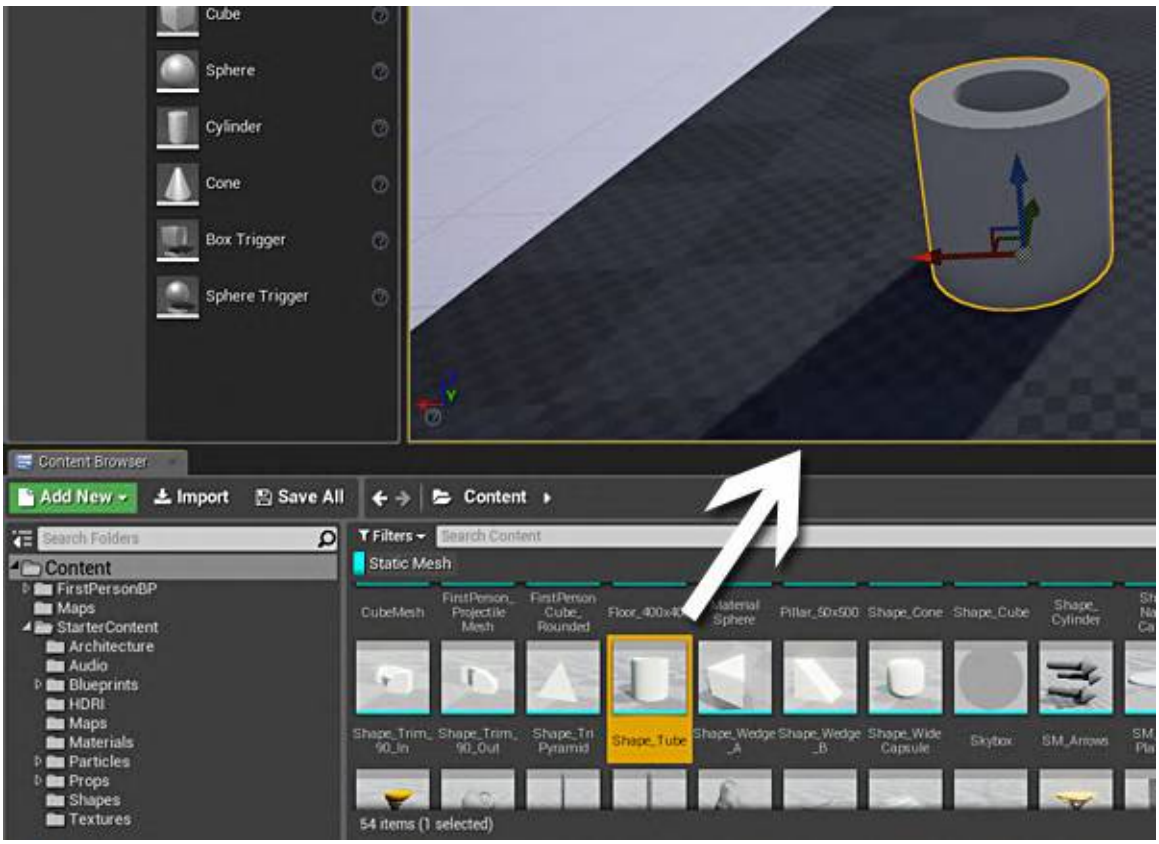
To use Static Meshes inside your level you will have to use the Content Browser. Look through the Content folder and any subfolder for the Static Mesh you need:



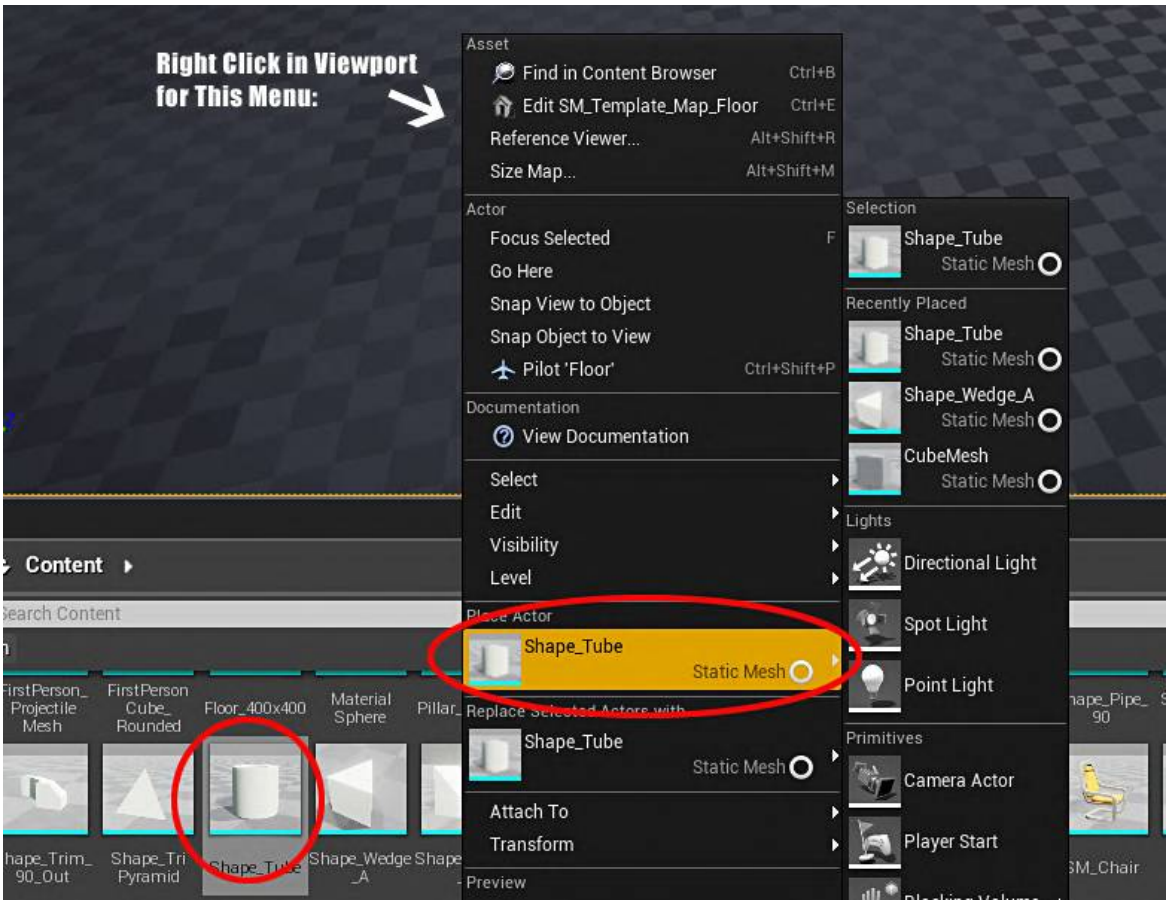
You can also filter by Static Mesh only:



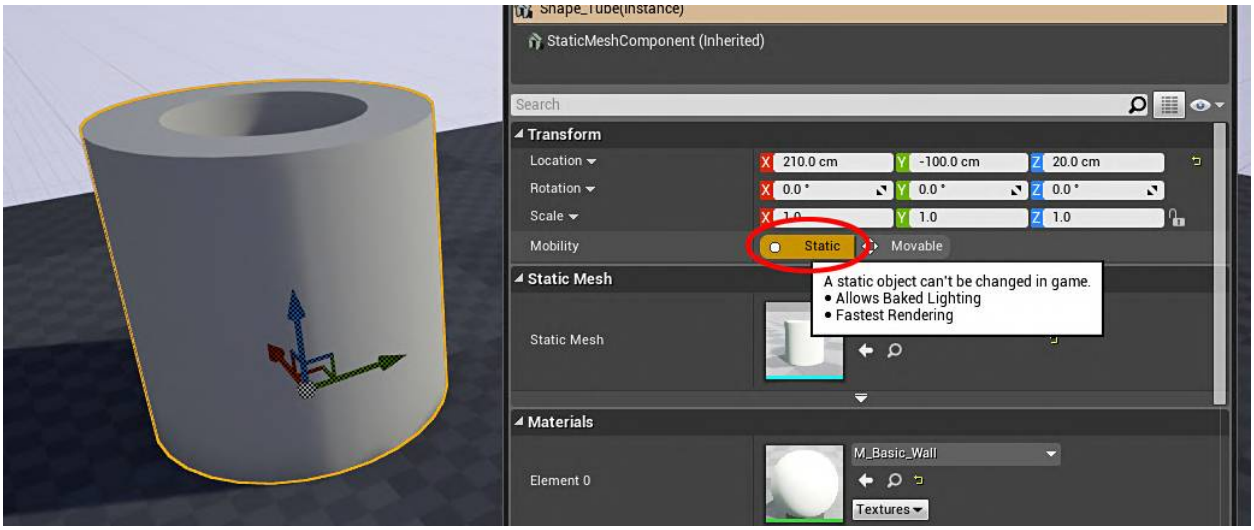
2 most common ways to insert Static Meshes into your level. Left-click and drag from the Content Browser into your level:



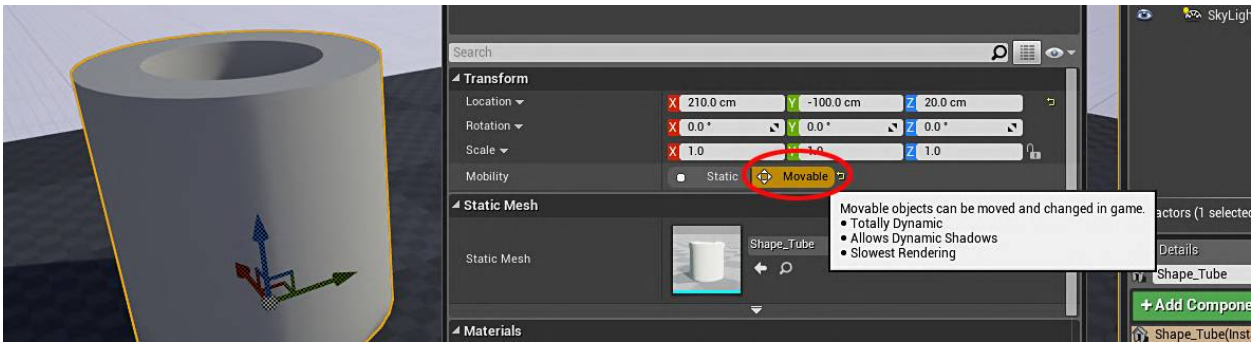
Select the Static Mesh in the Content Browser and inside perspective viewport, Right-Click and choose Place Actor:



By default, all meshes inserted into your level are static objects. Meaning they are fastest to render, they will use static baked in lighting and cannot be interacted with. This will be most of your Static Meshes:

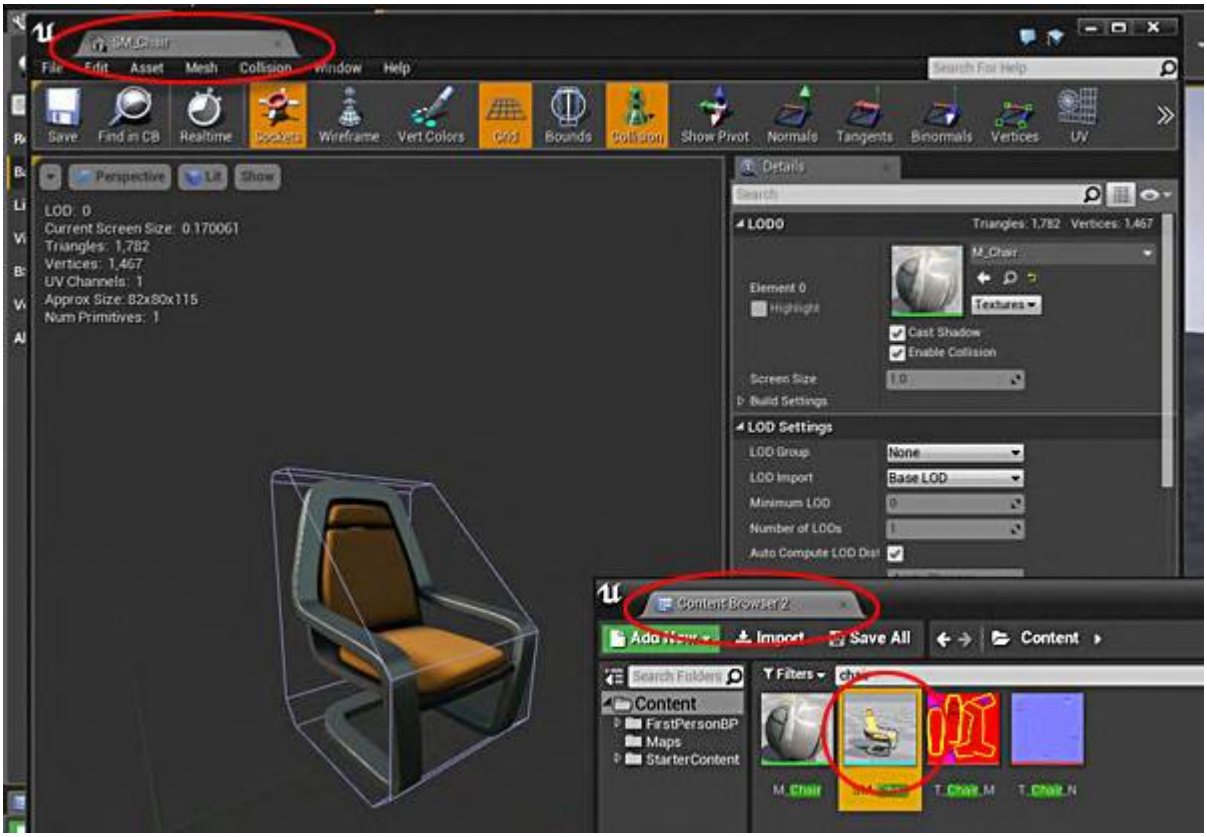


If you want a dynamic Static Mesh that can be used as an intractable object, use dynamic lighting/shadow then switch this mesh to Movable:



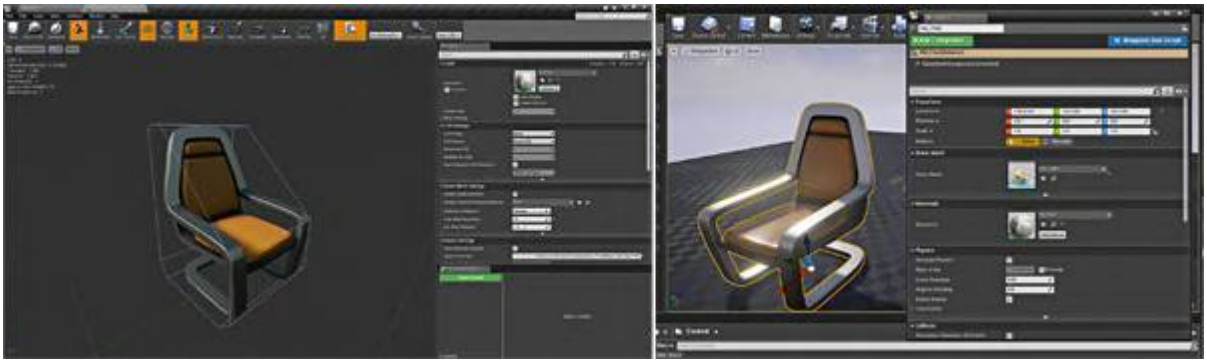
13. STATIC MESH EDITOR

Static Mesh Editor is the universal menu to edit a mesh. All changes will apply to every instance of that Static Mesh that is already placed or will be placed within your level. To access Static Mesh Editor, double click on any Static Mesh from the Content Browser:



Some of the properties inside Static Mesh Editor include Lightmap Settings, Level of Detail, Collision and Import.

14. STATIC MESH EDITOR VS STATIC MESH DETAILS SETTINGS



"When you do change Static Mesh Editor settings and when do you change Static Mesh settings through the Details panel?"

Let me give you a basic workflow of using Static Mesh Editor and Static Mesh Details panel:

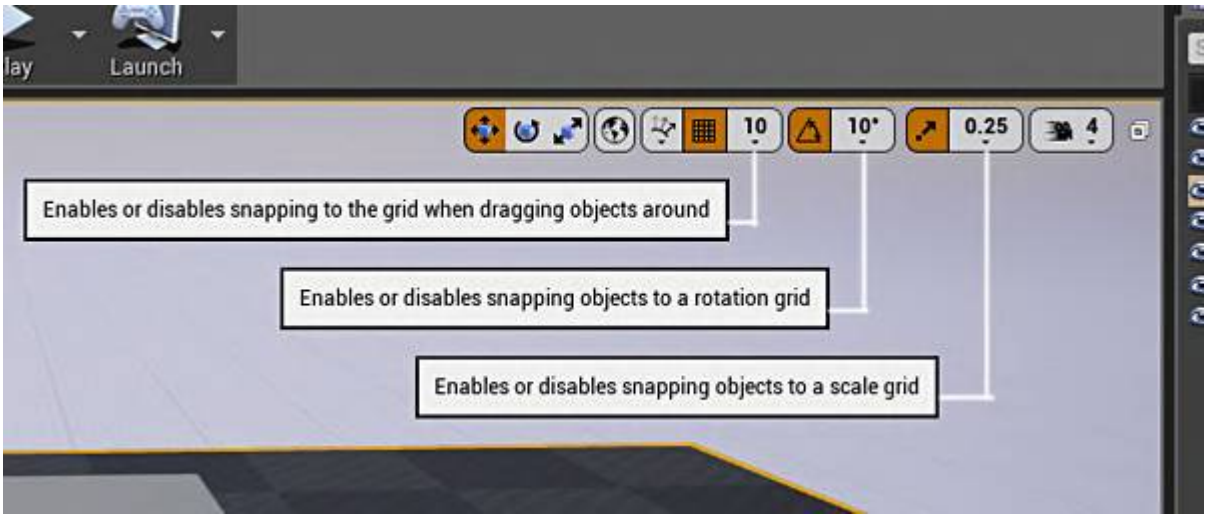
- If you want to update a setting to be applied to every instance of a Static Mesh across every single level that uses that Static Mesh - use Static Mesh Editor.
- When you import a Static mesh into UE4 for the first time you will use Static Mesh Editor to adjust all initial properties.
- If you want to change a specific setting for one Static Mesh placed within a level but not every duplicated instance, then select that Static Mesh within a level (already placed) and update properties through the Details panel.

15. USE GRID SNAPPING

Grid snapping is one of the most important aspects of constructing worlds**.** You want every BSP brush and Static Mesh you place inside your level to snap together like a set of Legos.

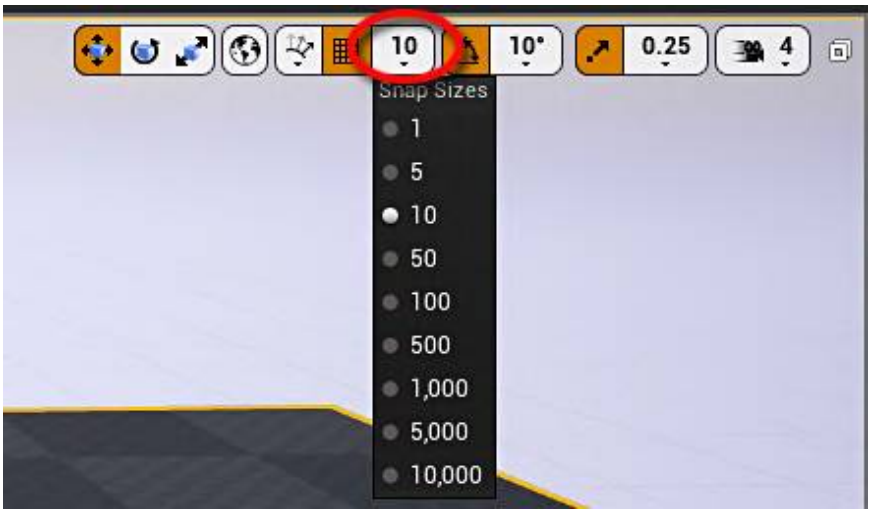
Grid snapping is enabled by default. To enable/disable grid snaps left-click on the move, rotate and scale snaps icon within the viewport:

You have 3 type of grid snapping options - move, rotate and scale.

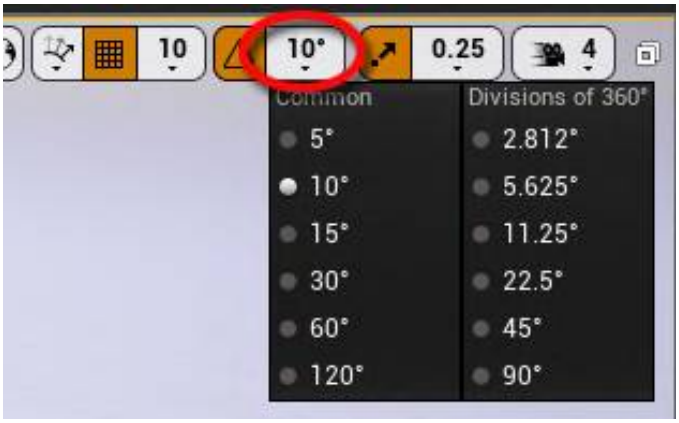

Launch
10
10°
0.25
4
Enables or disables snapping to the grid when dragging objects around
Enables or disables snapping objects to a rotation grid
Enables or disables snapping objects to a scale grid

Use the drop down menu to set the size of each grid type and at what values you want them to snap to.

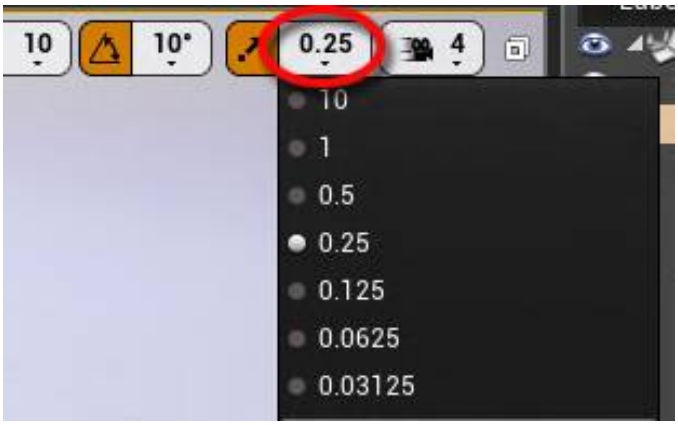
Move snap grid option will be one of the most important ones. Move grid spacing starts at 1 and continues 5, 10, 50, 100, 500, 1000, 5000, 10000:



Rotate snap degrees:



Scale snap values:



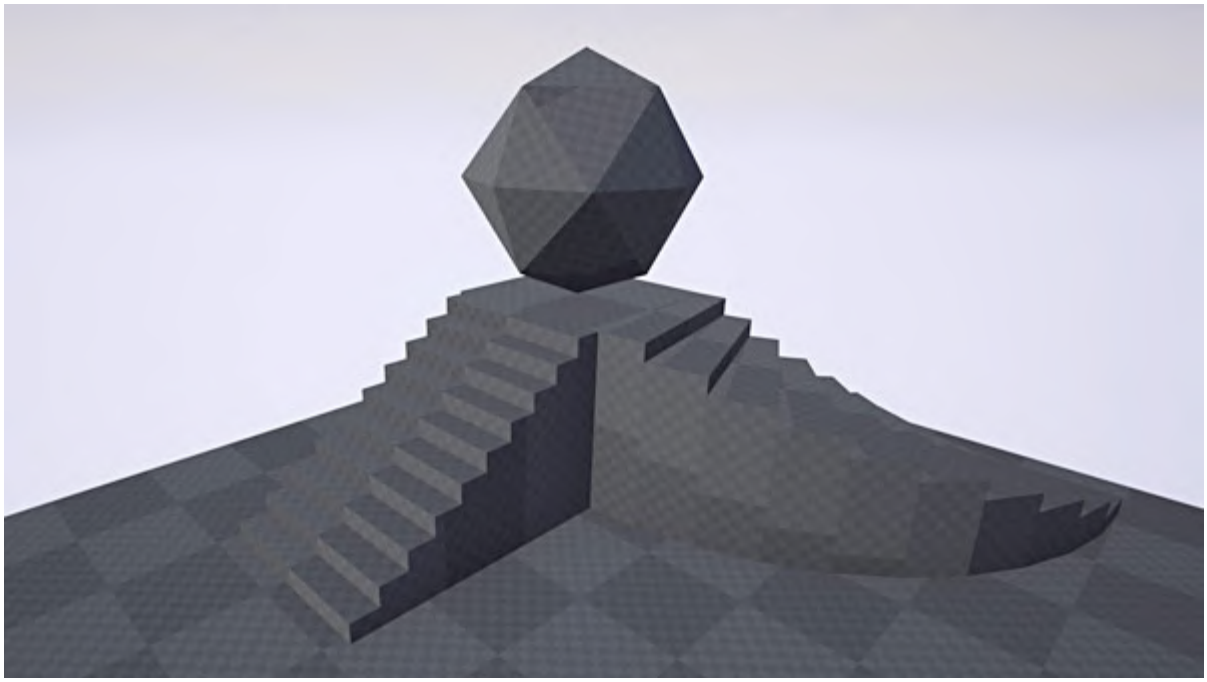
Shortcut key to changing move grid spacing are the bracket keys [].

- [ ] = Decrease/Increase Grid Size

Begin with grid spacing of 10, 50 or 100 as you construct the level. Switch to lower grid snaps of 1 and 5 for more precise detail work.

16. BSP BRUSHES

BSP brushes are the most basic building blocks of a level. They are simple geometric shapes that are created inside the UE4 editor.



So before you start detailing, lighting or using Static Meshes; BSP brushes will be your first go to for blocking in and prototyping a level.

Strength of using BSP brushes come from the ability to modify their shape as you work so you can iterate and test your level. As you use BSP brushes to block in your map, you will focus on the following:

- Layout
- Prototyping gameplay
- Gameplay testing your level
- Scale and proportion

The main key with BSP brushes is to create a shell of your level and to prototype how the level looks and plays before inserting any Static Meshes, working on any detailing or lighting.

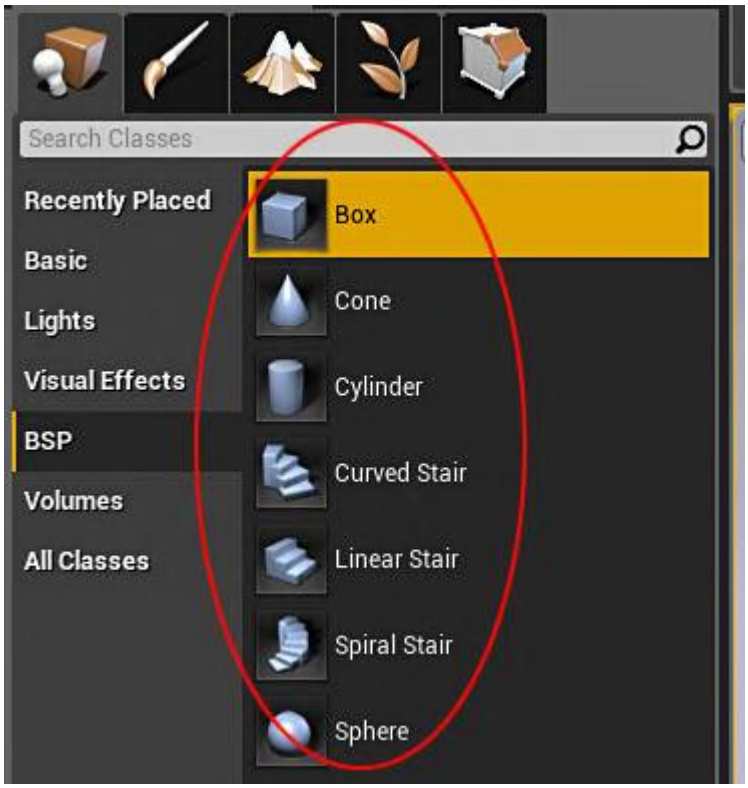
You will then replace all brushes with Static Meshes and detail your world.

17. ADDING AND SUBTRACTING BSP BRUSHES

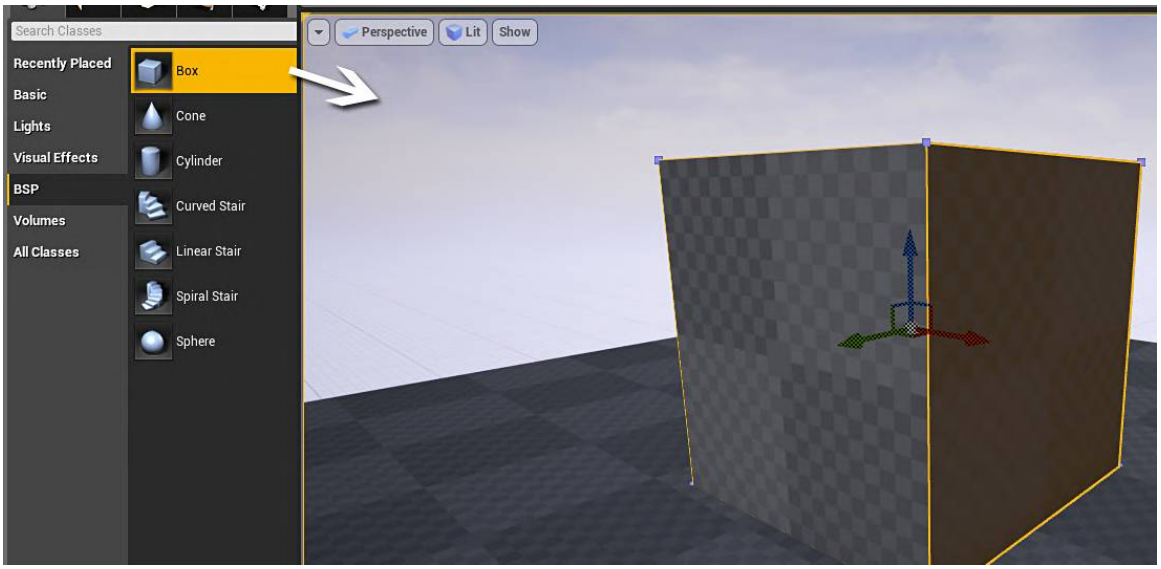
Basic BSP brush workflow is to Add and Subtract them from your level.

To include your first BSP brush, use the Place Mode Panel (Shift+1) and switch to BSP tab:

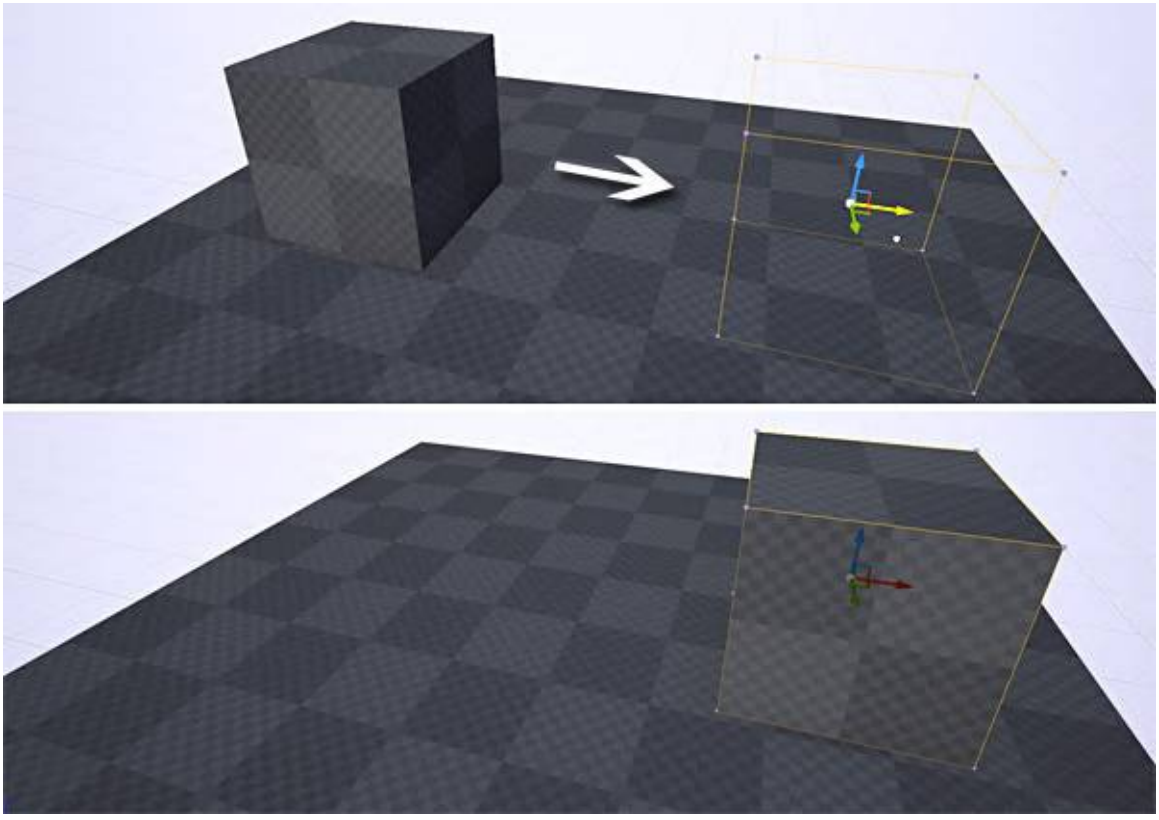
You can choose from a set of various geometric shapes such as a Box, Cylinder, Stairs or Sphere:



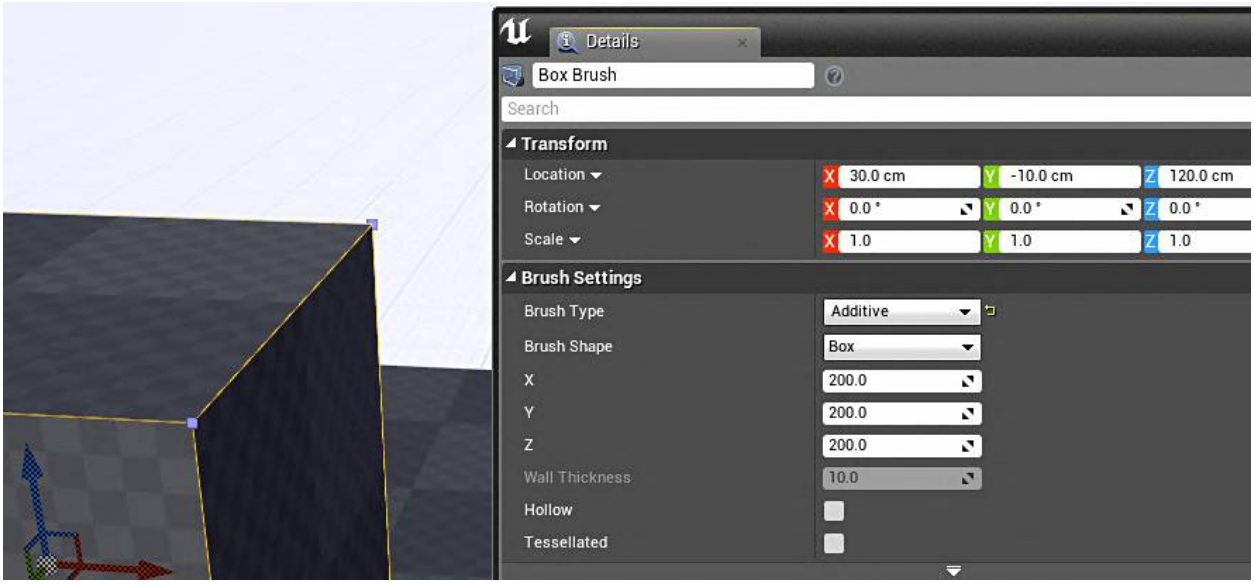
Left click and drag a brush shape that you want into your perspective viewport:



In perspective viewport, select the brush and use the Spacebar to cycle between move/rotate/scale transformation gizmos. Anytime you move these brushes within the level, they will automatically update:



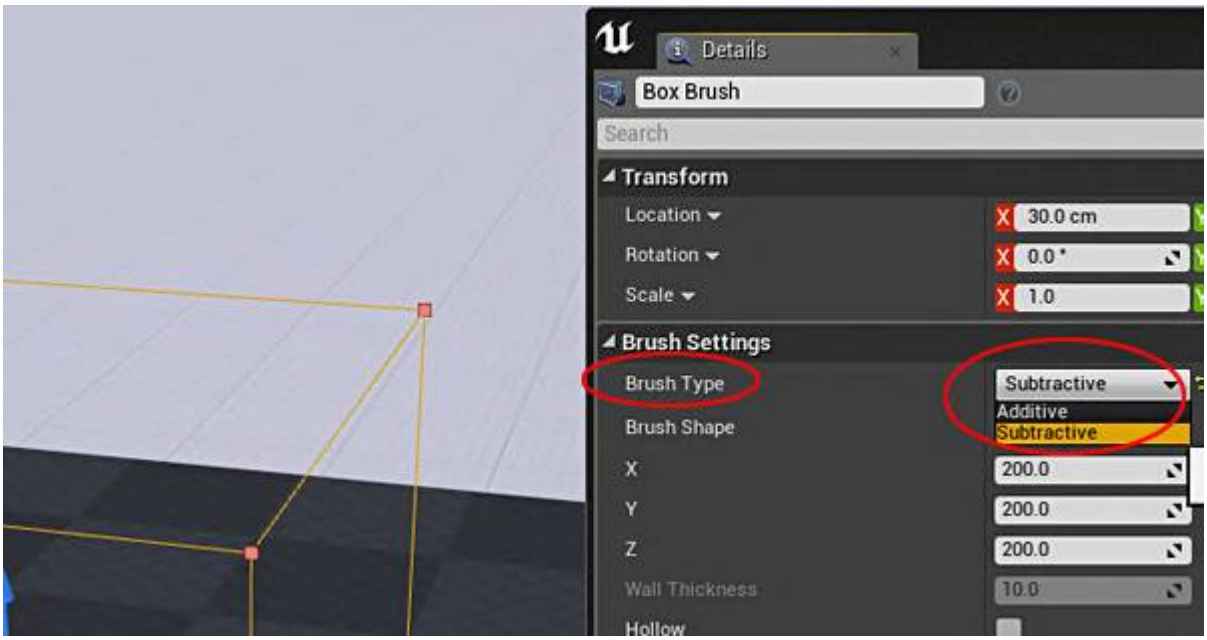
Selecting any BSP brush will enable the Details panel for that brush:



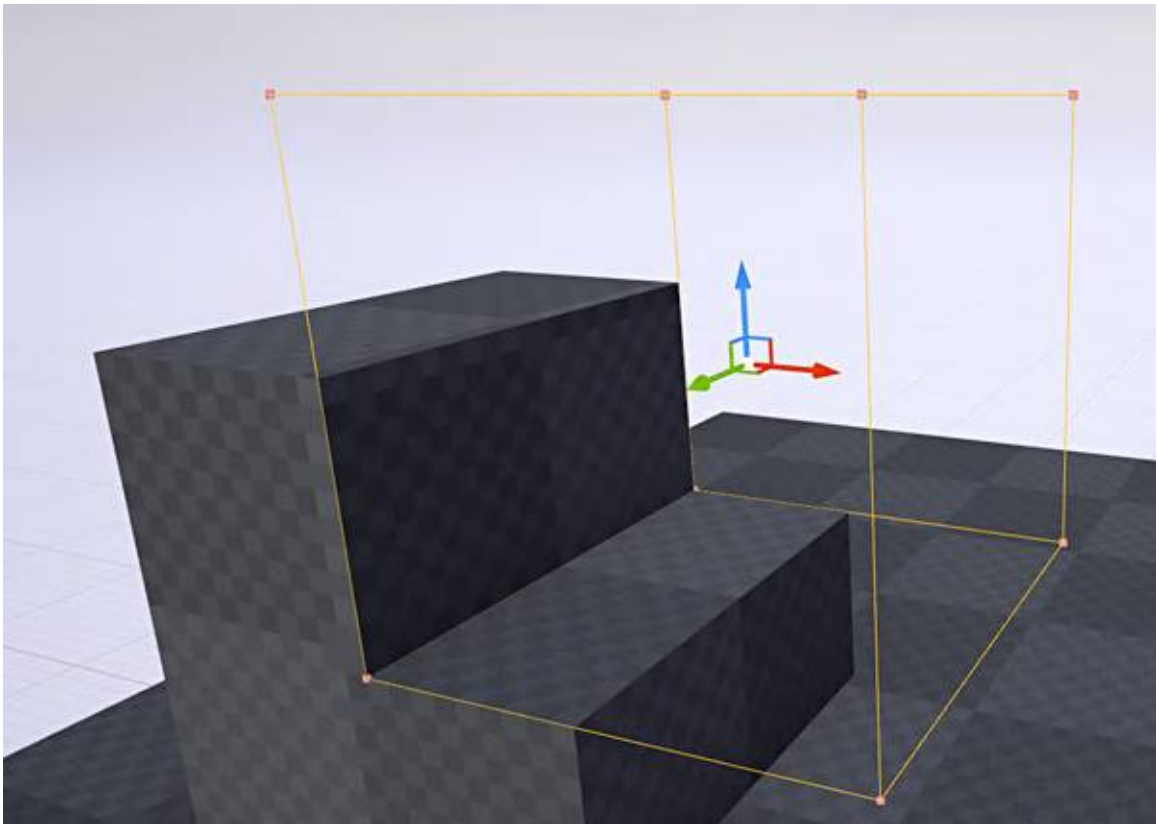
Through this Details panel you can adjust various settings for that brush. Such as changing brush type from additive to subtractive, changing the size of the brush and managing materials placed onto that brush:

Once you have added a brush into your level, you can subtract from it.

To subtract from already placed/added brushes re-drag a BSP brush and in the details panel change it to subtract function:

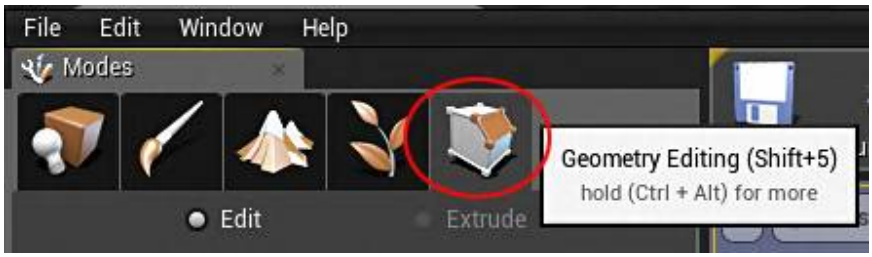


Then drag the subtractive brush over the additive brush and where these brushes overlap, that geometry will be removed:

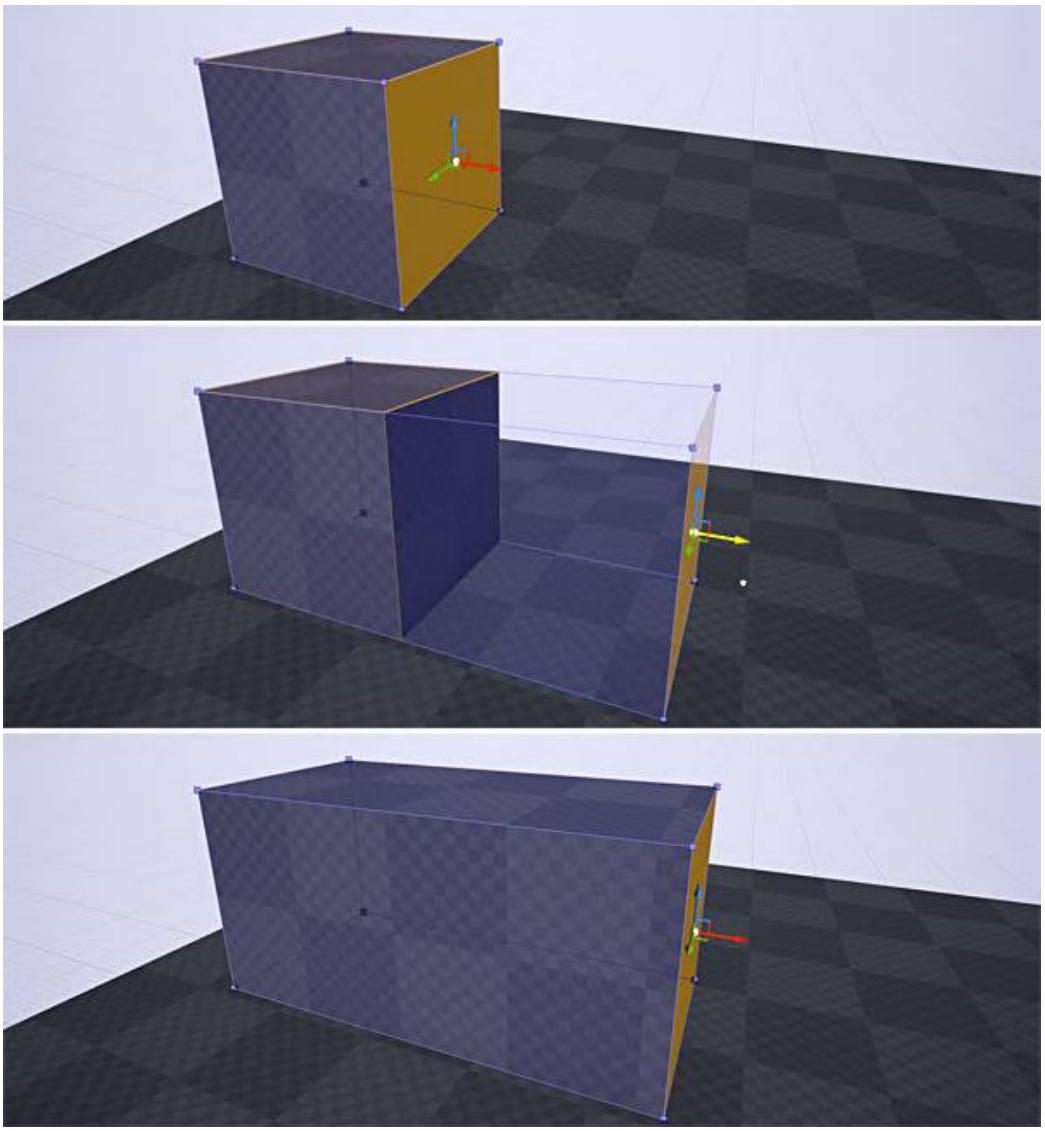


18. USING GEOMETRY EDITING MODE

Select a BSP Brush and press Shift+5 to switch to Geometry Editing Mode:



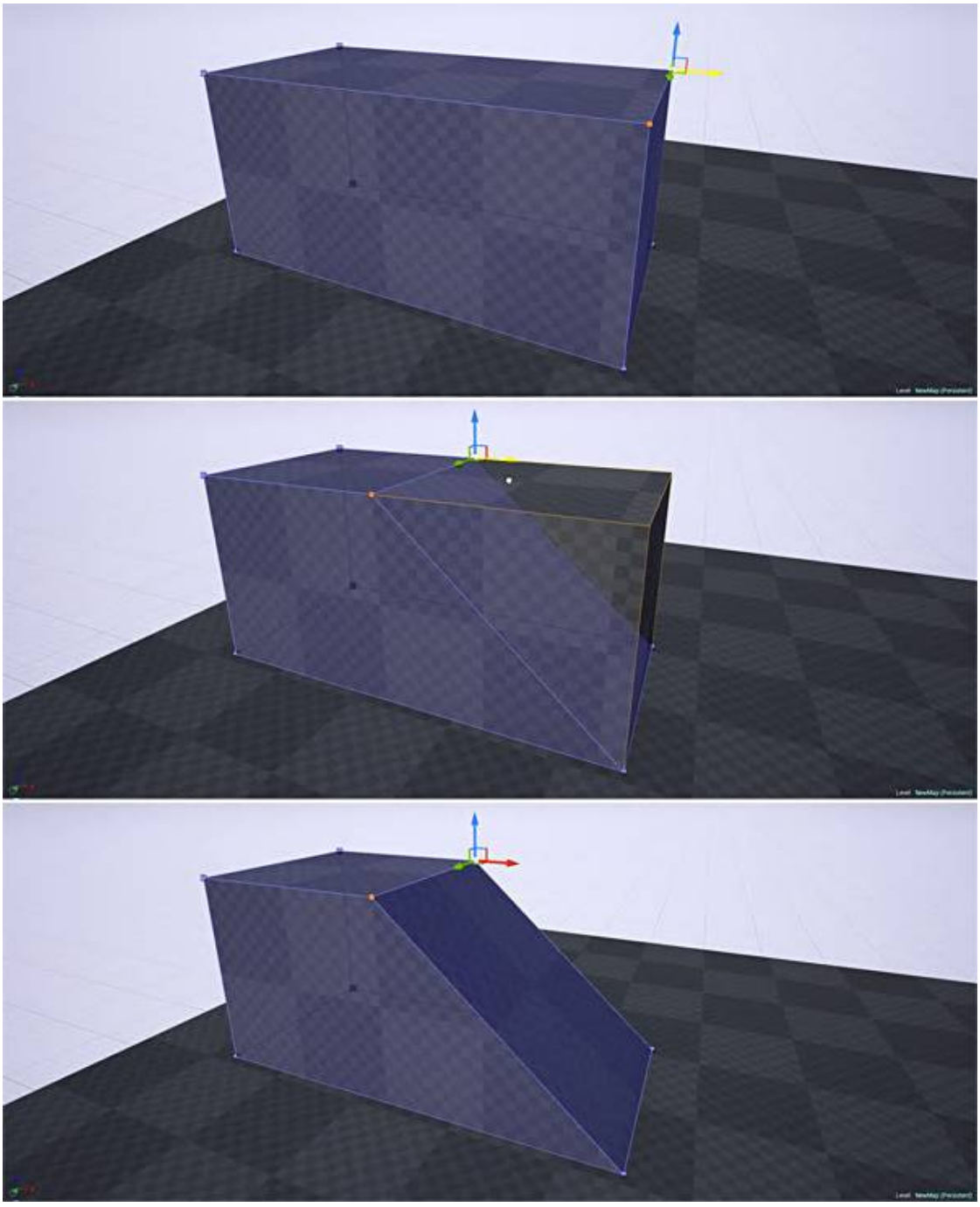
You can then select vertices, edges and faces of any BSP brush and quickly modify their appearance:



Instead of changing the size of the brush via Details panel, you have faster iteration and control of the brush by using Geometry Editing (Shift+5).

Select the BSP brush, press Shift+5 for Geometry Editing mode and left-click to select any component (vertices, faces or edges).

Now just drag that component and the brush will change the shape to what you want:



Few important considerations:

- Do not modify a single vertex. Meaning, don't select and drag one vertex. This will create abnormal brush shapes.
- Keep BSP brushes as simple as possible; remember these brushes are just place holder and will be replaced later with Static Meshes.
- Duplicate BSP brushes by switching to Place Mode (Shift+1); select the brush, press Alt+Move the brush to duplicate a copy.
- You can also use the geometry editing mode set to extrude, by dragging a face it will extrude a new segment from it. Quick way to get more complex shapes.

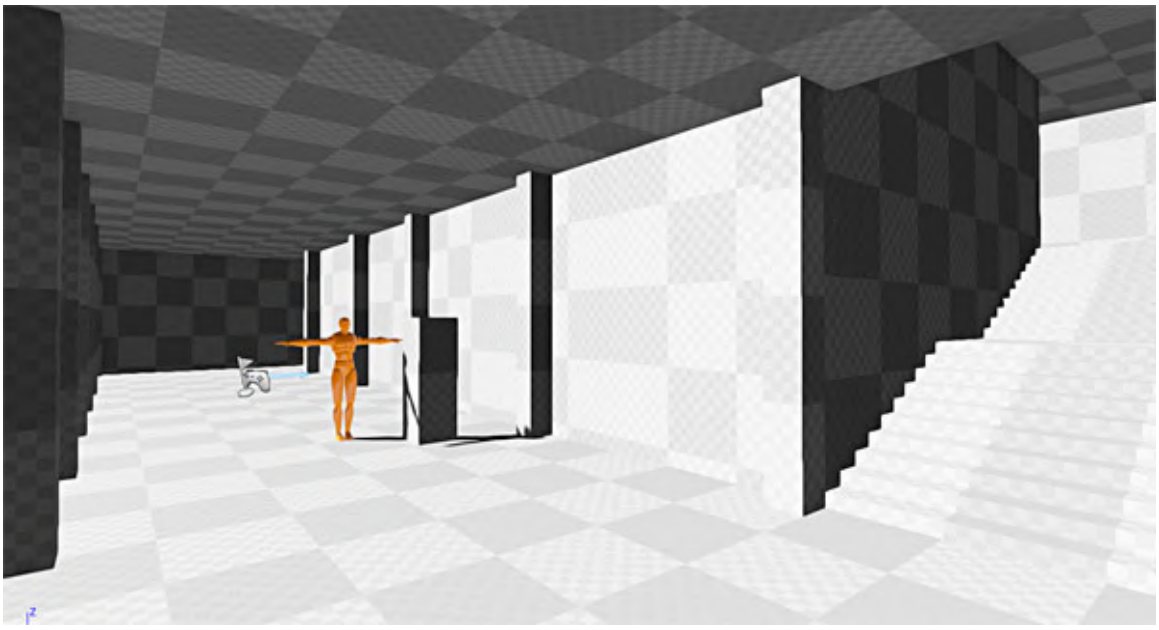
19. BSP VS STATIC MESHES

"How much BSP should be used in the final level compared to Static Meshes?"

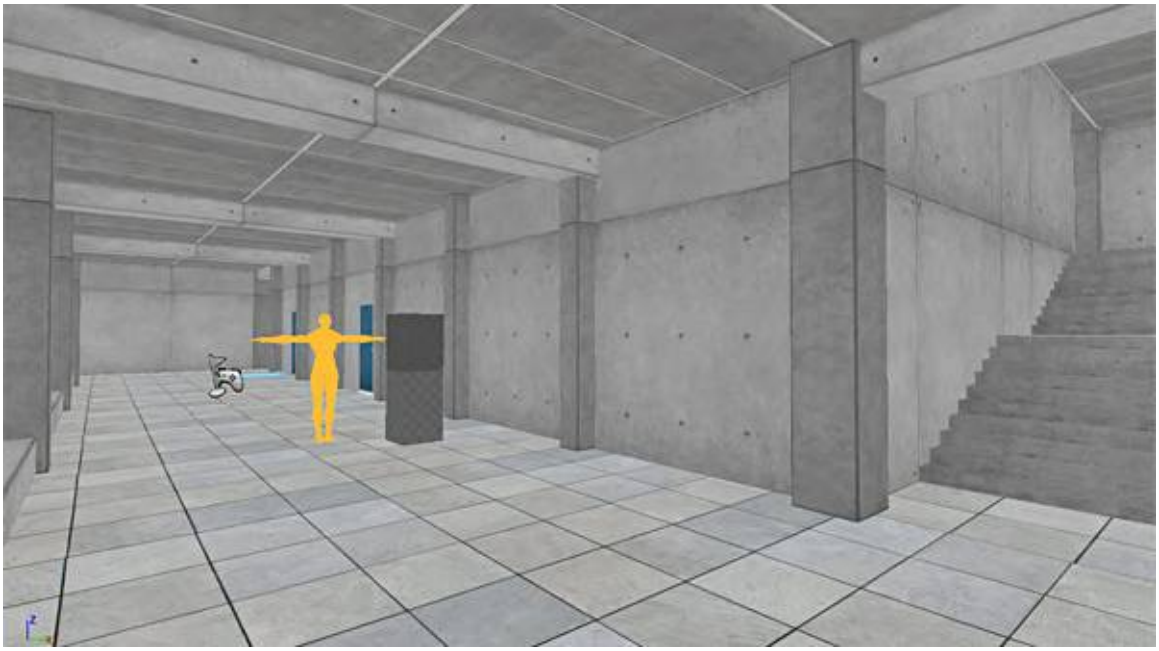
Final level should contain as little BSP as you can get away with. It is very common to have 100% all Static Meshes. But 90/10, 80/20 or 70/30 rations of Static Meshes to BSP are also acceptable.

Your map can contain BSP brushes but their use should be minimized as much as possible.

BSP only:



Static Meshes replacing BSP:



20. PLAYER SCALE AND PROPORTION

Scale and proportion is probably one of the most important elements when building levels in UE4 or any other game engine.

Everything you create must stay true to correct player scale and proportion. Nothing destroys the illusion of your level faster than disproportionate game world.

4 keys to build everything to correct scale and proportion:

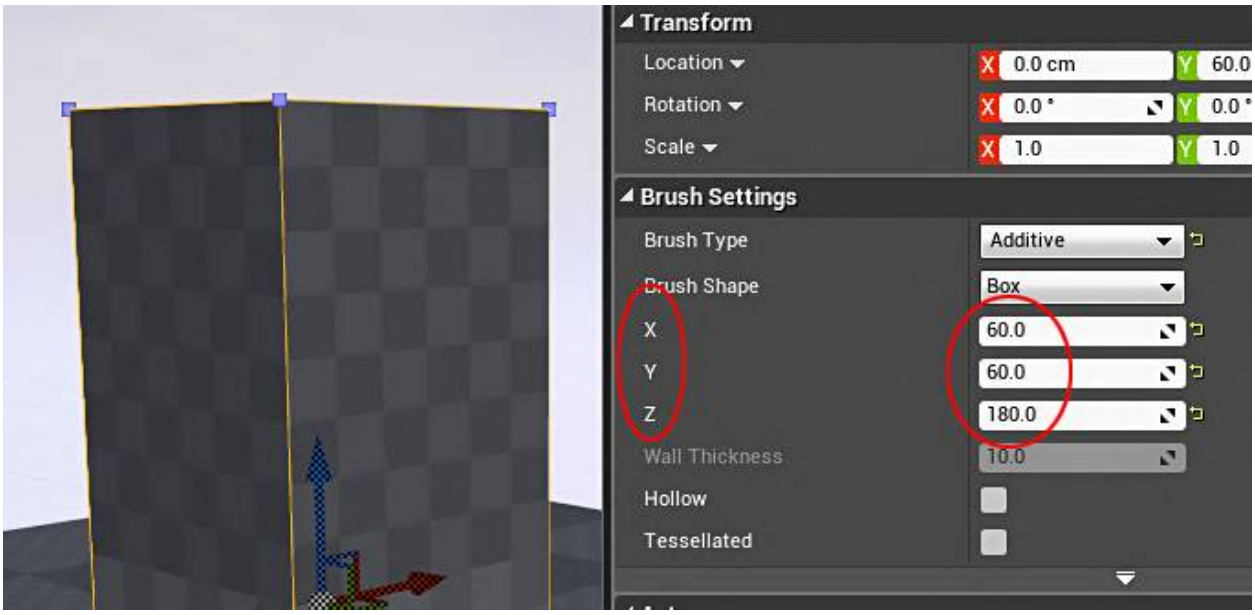
- Insert a character scale model that is the same dimensions as a player inside the game
- Establish base dimensions for your architecture
- Reference your construction to player character scale and architecture surrounding it
- Play test as often as possible from the game itself

In Unreal Engine 4, base character scale to start with is:

- Height: 180uu/cm
- Width/Length: 60uu/cm

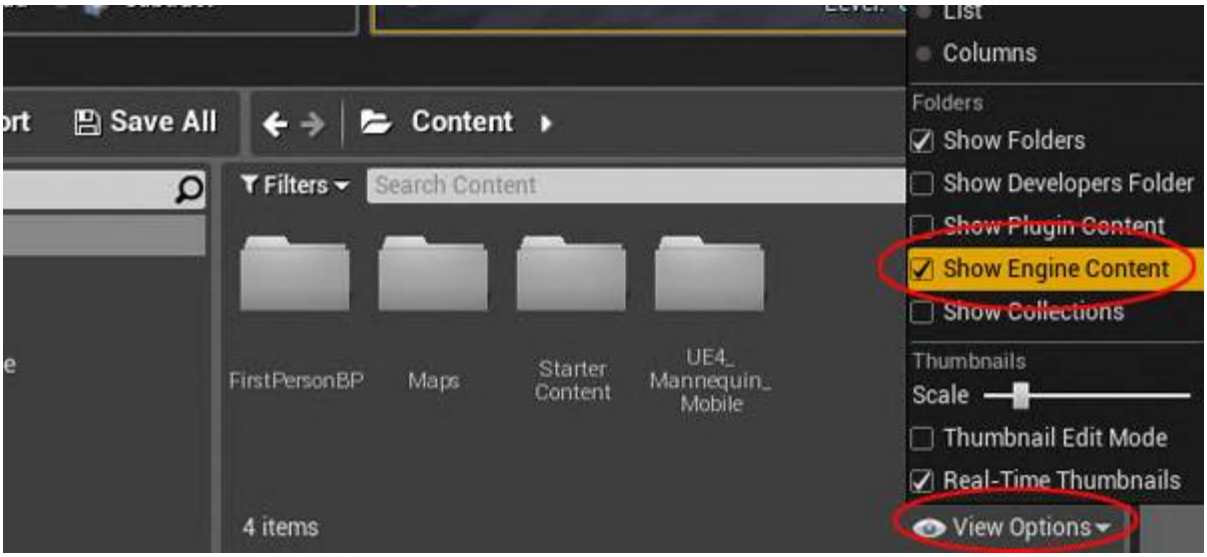
Scale and proportion is always relevant to the game you are working on. 180uu height for a character is a starting point, but for a specific game and its world these values might be different. Always reference the world to the character scale that is going to be used in your game. Character scale can be a simple box or a skeletal mesh that contains the same dimensions as the player character in the game.

Option A: Create a BSP brush with the following player scale dimensions:

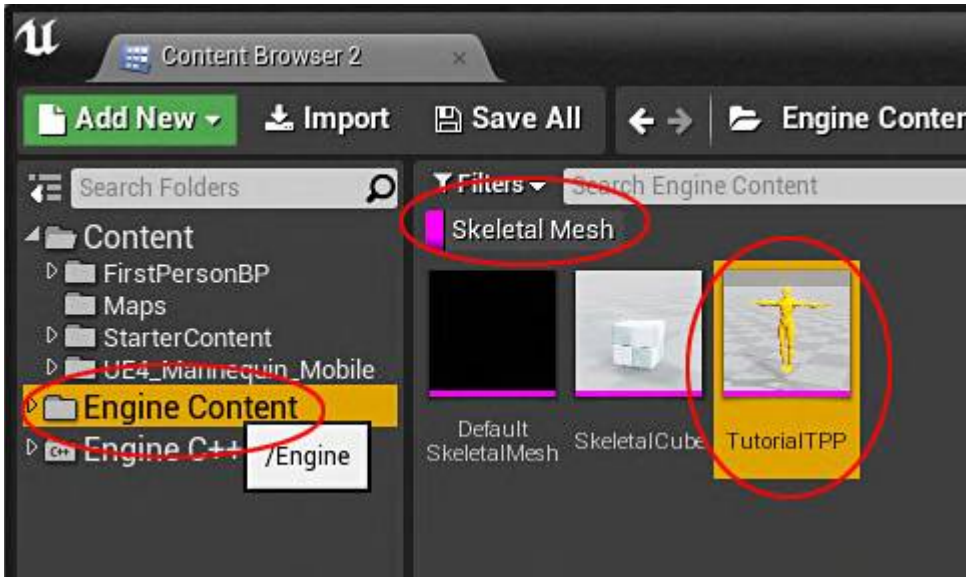


- Height: 180uu/cm
- Width/Length: 60uu/cm

Option B: Insert a skeletal mesh reference from the Epic's Content Folder. Go to Content Browser and under View Options enable "Show Engine Content":



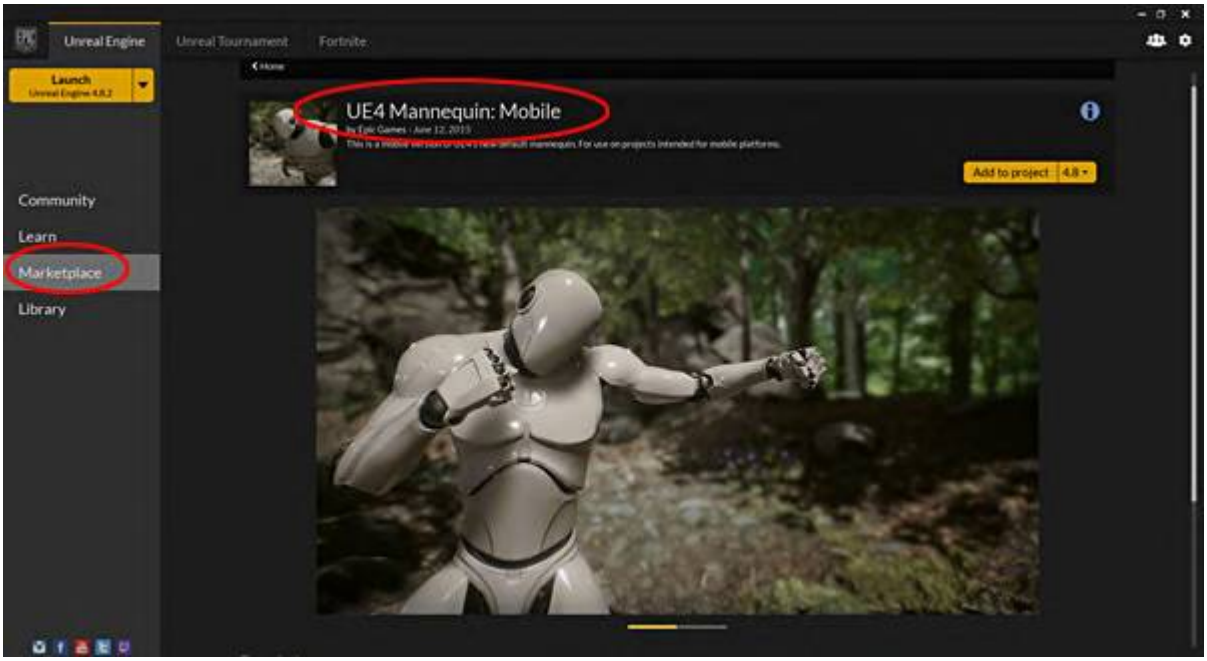
Select Engine Content folder and filter by Skeletal Mesh:

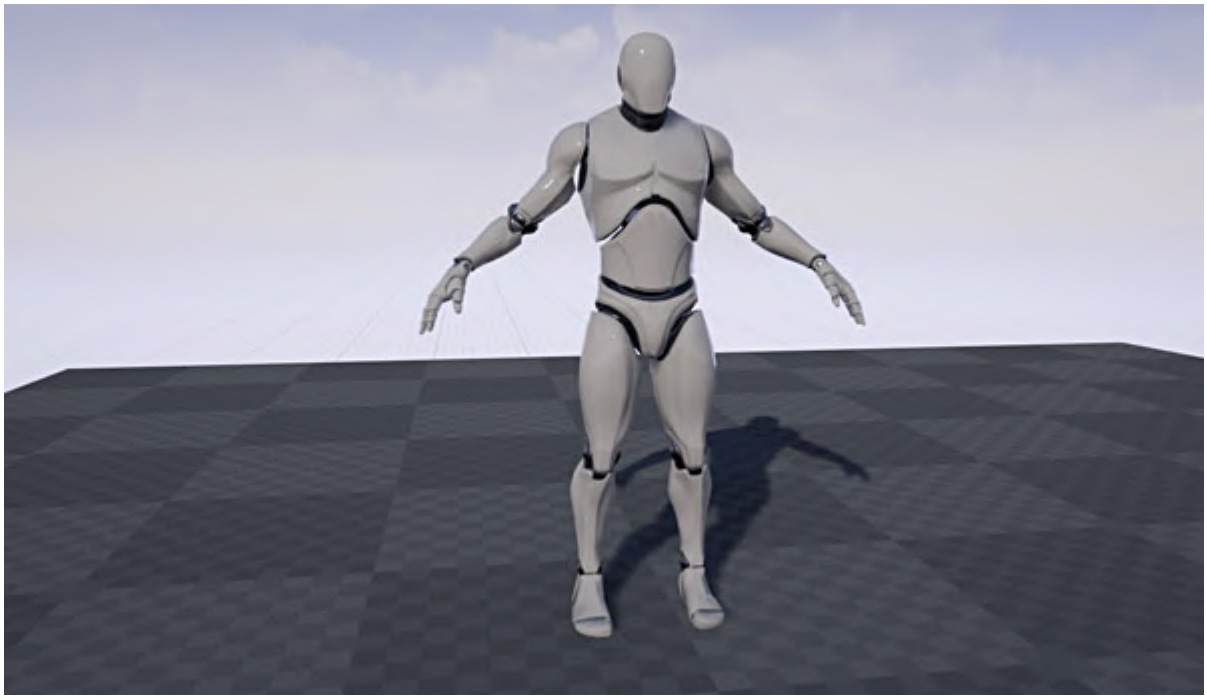


Use the TutorialTPP skeletal mesh character as a starting point for your scale.

This character is slightly more than 180 units in height, so scale this down using the Details panel on Z axis down to .94:

Option C: Download Epic Game's free low poly skeletal mesh character from the Marketplace under Characters and Animation category:





Option D: If you creating or working on a game, check to see what scale and dimension the game uses and provided character mesh as a reference. If you are creating a stand-alone game environment without any gameplay, then use 180uu/cm height as the character guide.

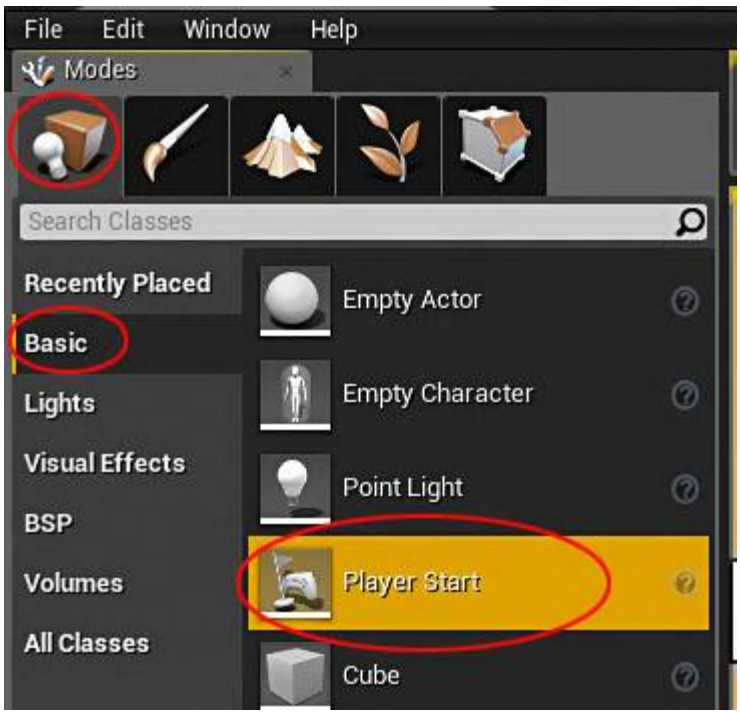
Insert one of these options into your map throughout the entire environment to judge scale and proportion as you build. Once you have a layout and everything is proportionately correct, you can delete these character references from your level.

21. INSERTING A PLAYER START

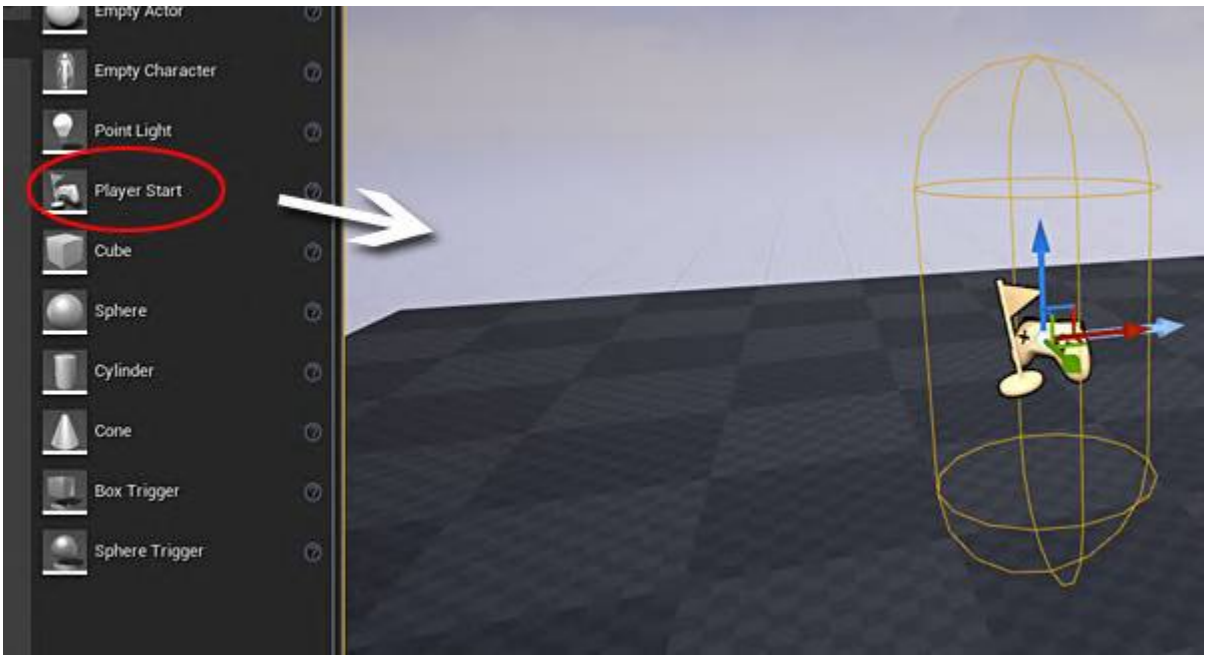
Player start defines where a character player will spawn from inside the level. Every environment should have at least one player start.

If you are creating a stand-alone game environment that doesn't require player participation, then you probably won't need one. Although, it is a good habit to always insert one player start into your level.

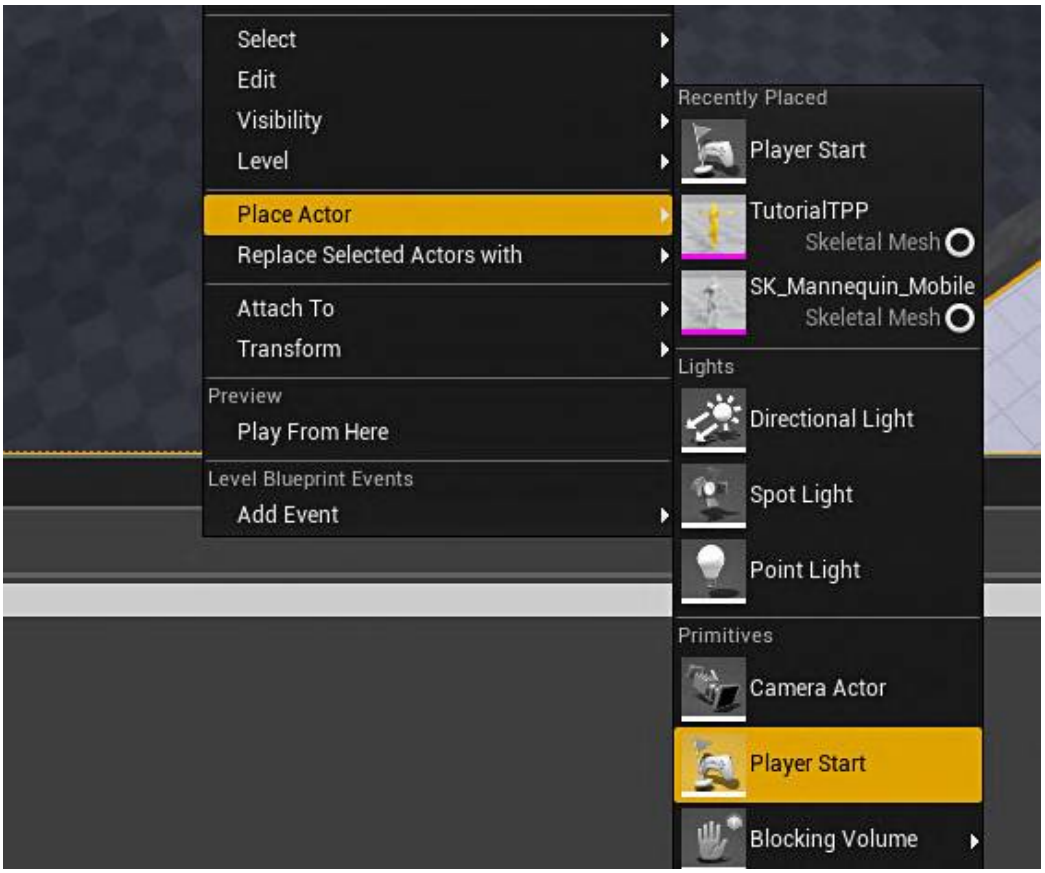
To insert a player start, go to Place Mode (Shift+1), switch to Basic:



Left Mouse Click and drag Player Start from the menu right into your level:



Another common way to insert is Right Click inside perspective viewport, choose Place Actor > Player Start:

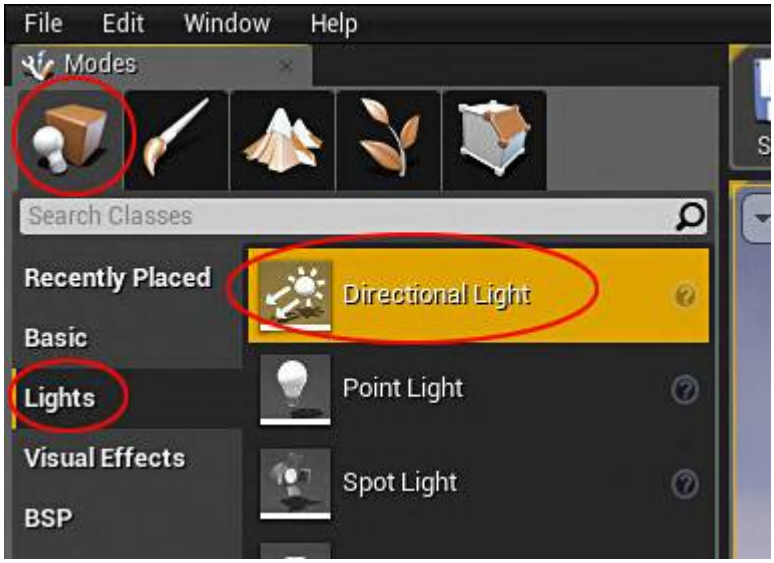


22. ADDING DIRECTIONAL LIGHT (SUN LIGHT)

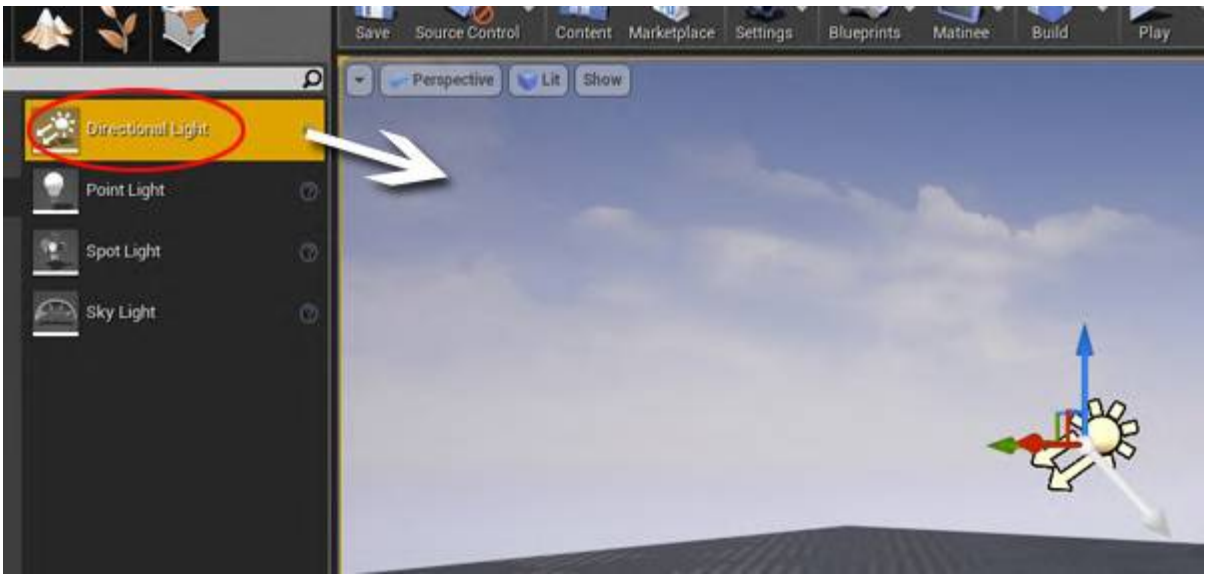
Directional Light is a light actor used for lighting exterior environments. Think of a Directional Light as a sunlight or moonlight:

You only need one of these inside your map and it will light your entire world (as a sun would).

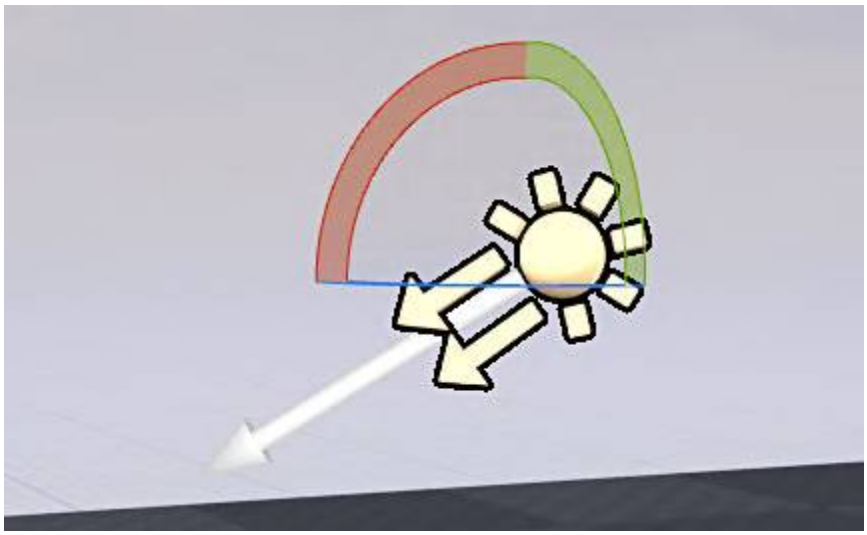
Place a Directional Light by going to Place Mode (Shift+1), switch to Lights tab:



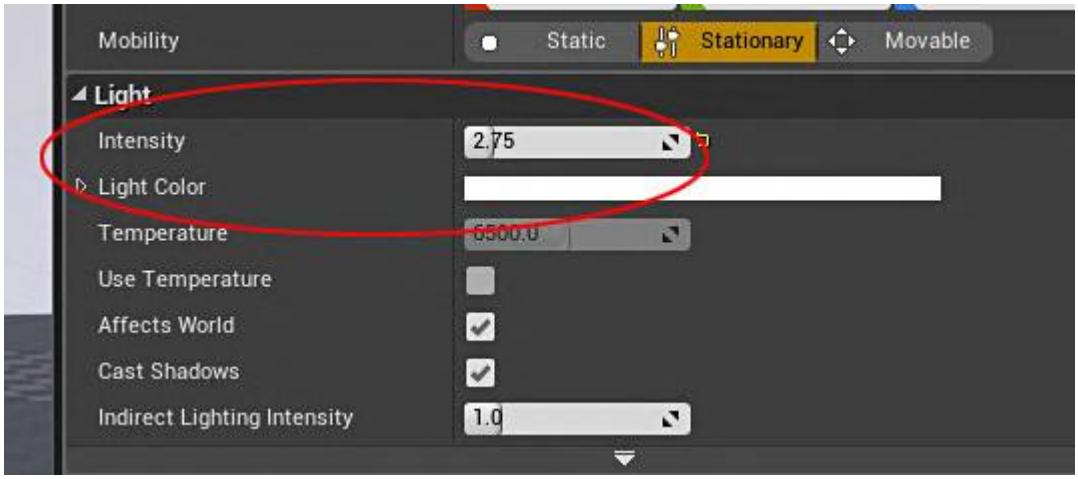
Left Mouse Click and Drag Directional Light into perspective viewport:



Rotate (E) the Directional Light inside your level to point the direction and angle you want:



Select the Directional Light and in Details panel change these two initial properties:



- Light color: color of the light
- Intensity: strength of the light

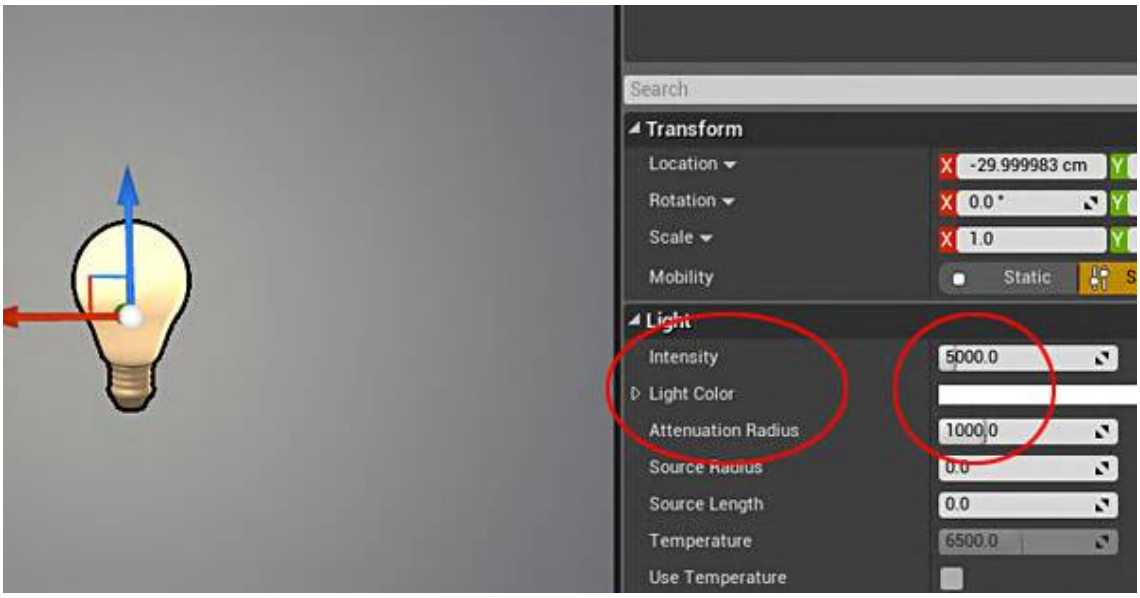
23. ADDING POINT LIGHT OR SPOT LIGHT

Point Lights and Spot Lights are most used lights for interiors or specific areas within a level. Both of these lights would be coming from a physical source such as a lamp or a street light.

To place Point Light or Spot Light into your level, go to Place Mode (Shift+1), switch to Lights tab:

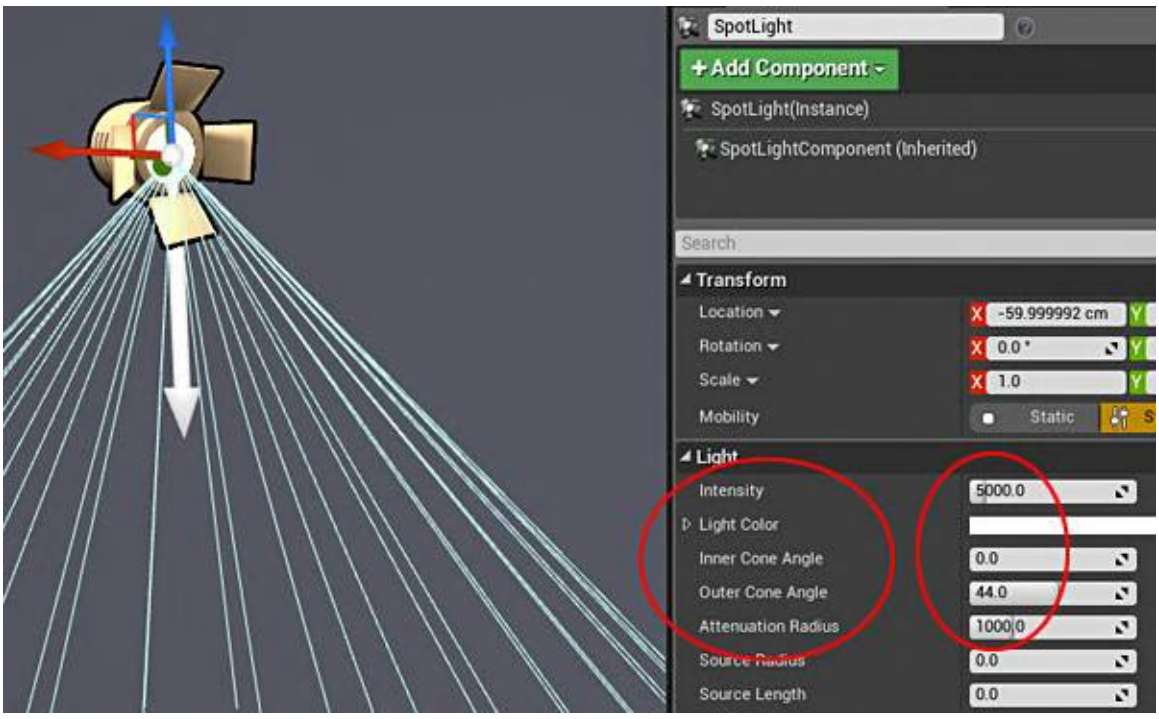
Left Mouse Click and Drag a Point Light or a Spot Light into perspective viewport of your level.

Point Light: After inserting the Point Light, select it and in Details panel start with the following properties:



- Light color: color of the light
- Intensity: strength of the light
- Attenuation Radius: lights area of influence

SpotLight: After inserting the Spot Light, select it and in Details panel start with the following properties:

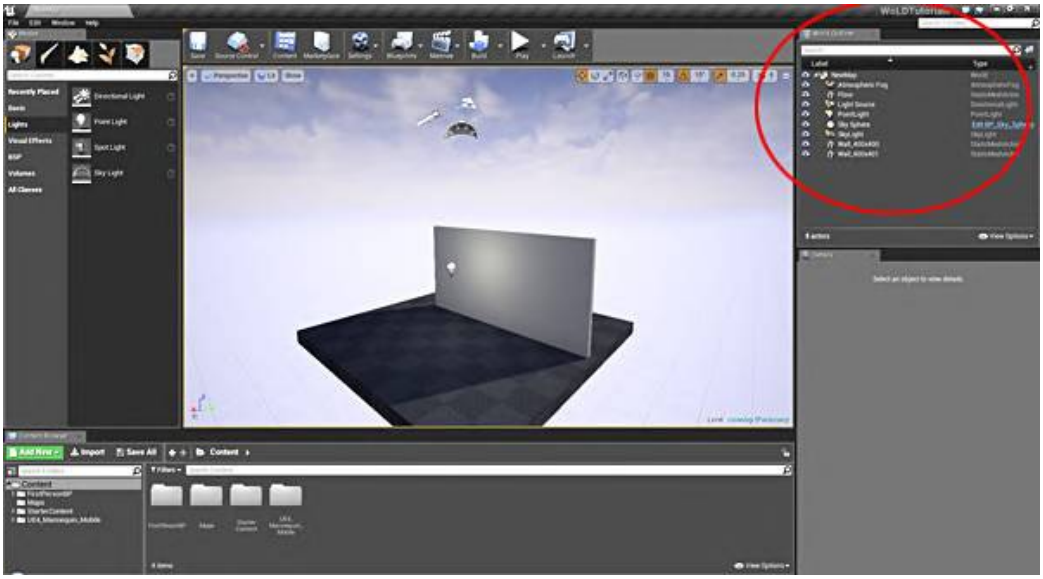


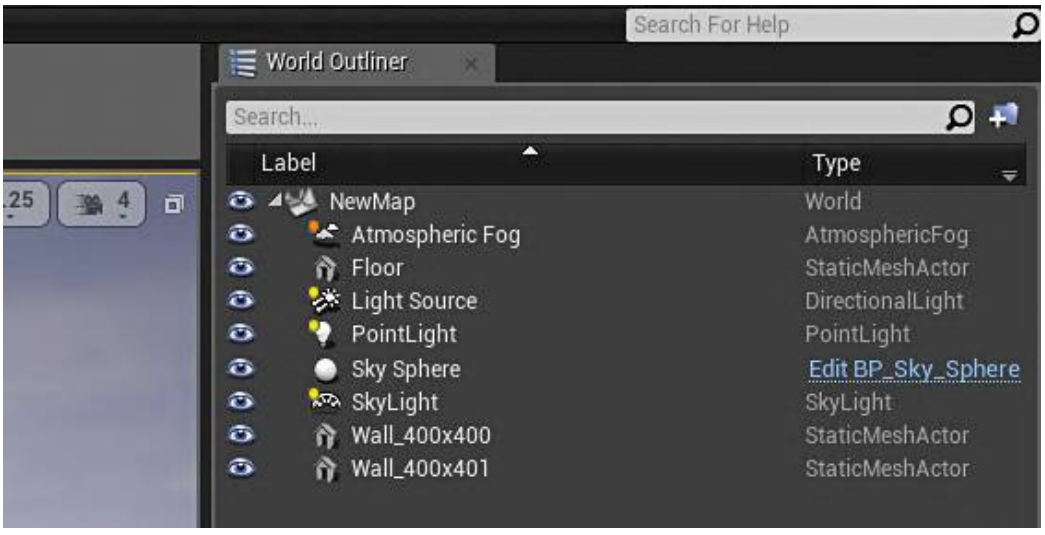
- Light color: color of the light
- Intensity: strength of the light
- Attenuation Radius: lights area of influence
- Inner Cone Angle: set the amount in degrees for inner angle
- Outer Cone Angle: set the amount in degrees for outer angle

All the properties you change will be visible in perspective viewport.

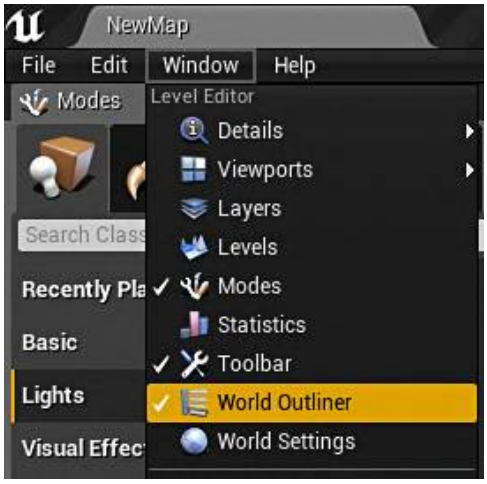
24. WORLD OUTLINER

World Outliner will list all actors/objects within your level. It is located on upper right hand side of the editor:



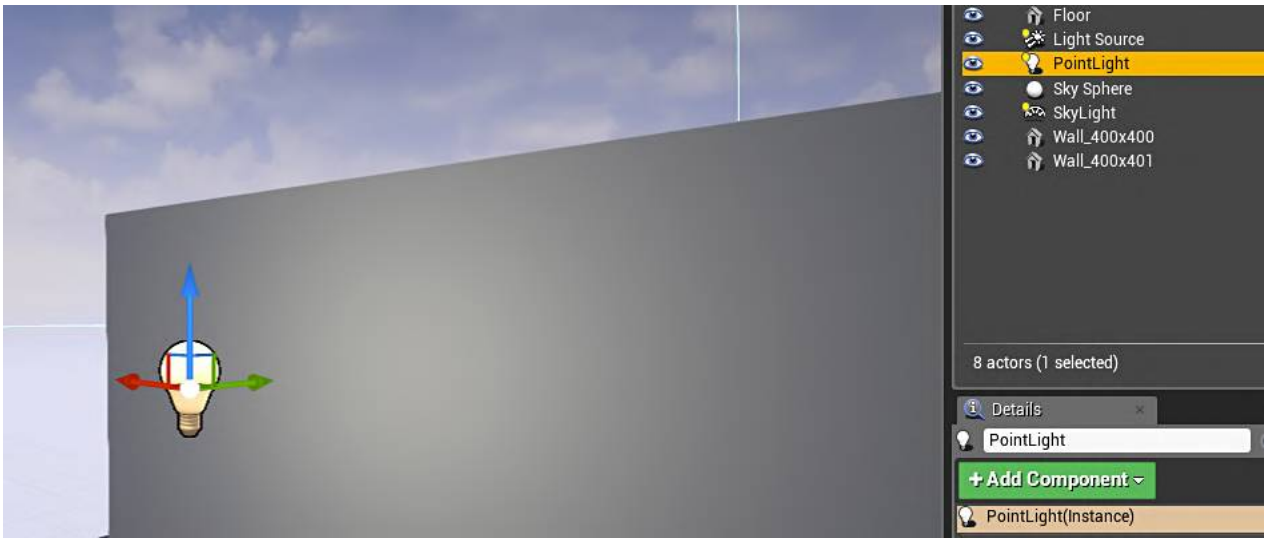


If you do not see it, go to Window > World Outliner:

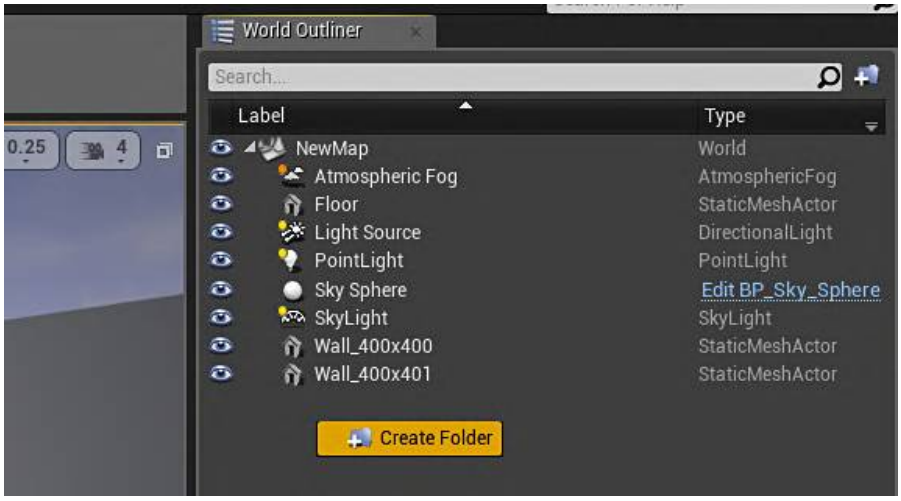


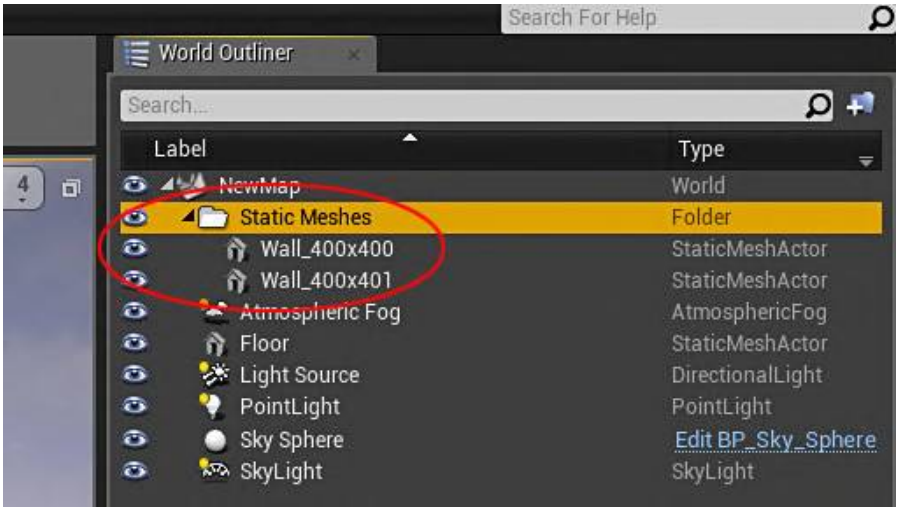
Using the World Outliner, you can search for any actors inside the level:

You can double click on any actor from World Outliner to center the viewport on that object:

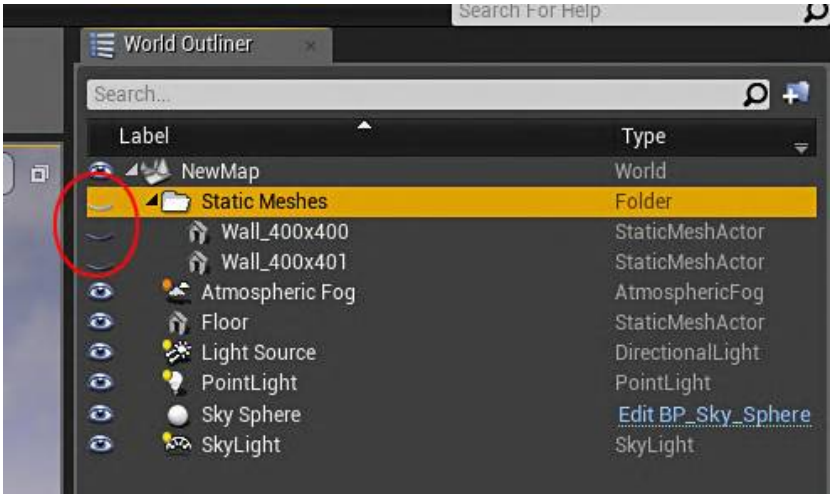


And, you can organize your scene better by using folders:



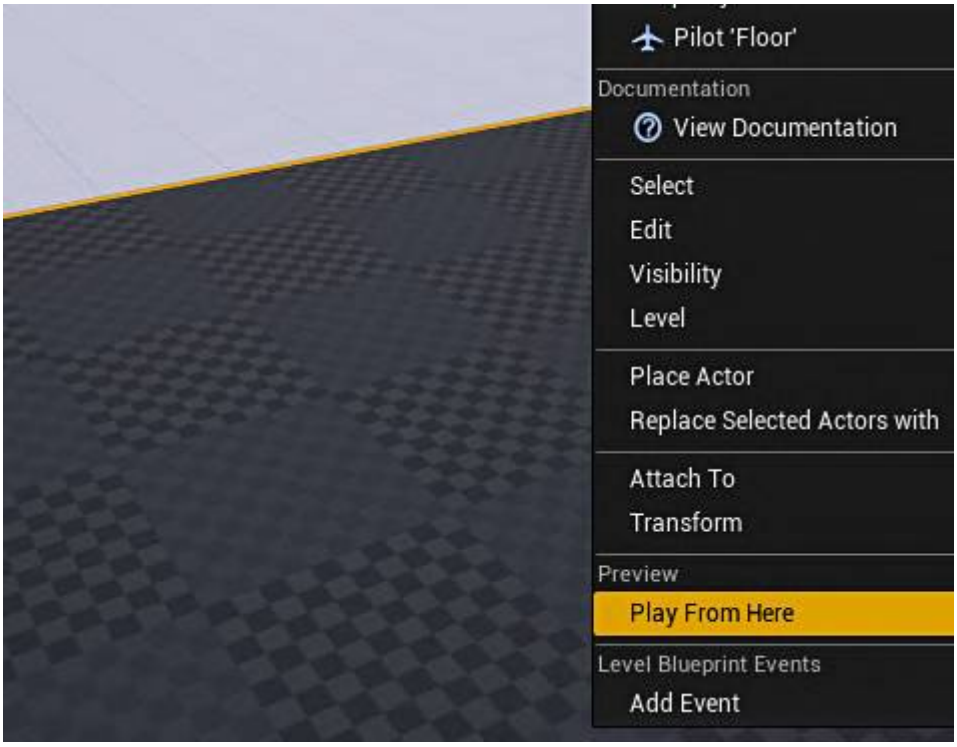


And visibility option:

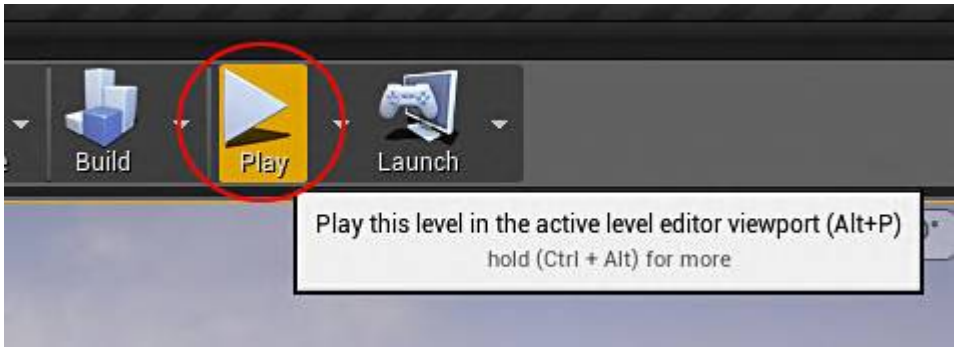


25. PLAY TEST YOUR LEVEL

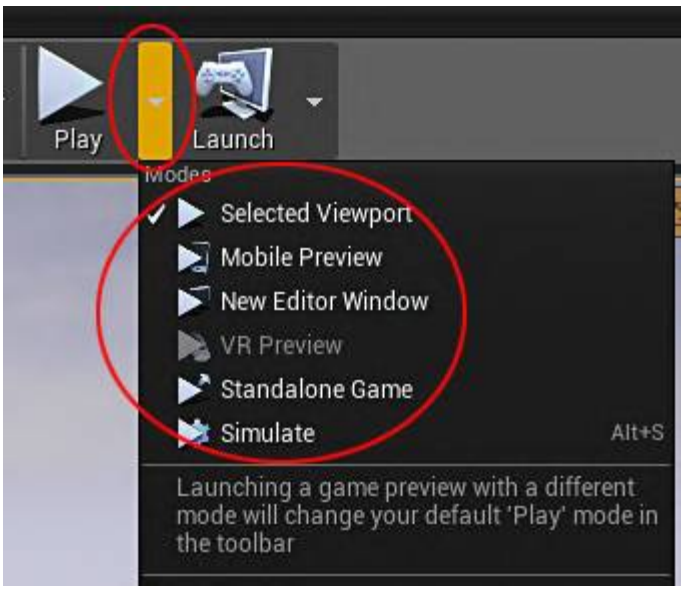
You can play test your level as a character inside the game right from the editor. In perspective viewport, Right Click where you would like to spawn from and choose “Play From Here”:



To use a Player Start as a spawning location, click on Play icon at the top toolbar:



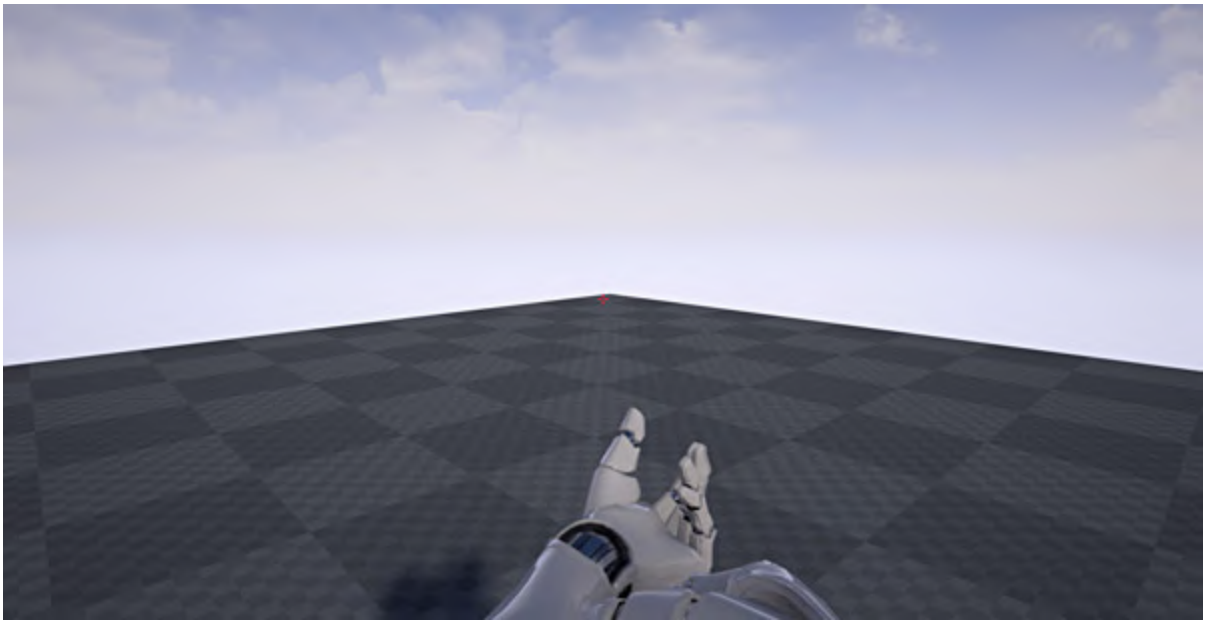
For additional options use the drop down menu and choose how you want to Play Test your level; such as "Selected Viewport", Mobile Preview or New Editor Window:



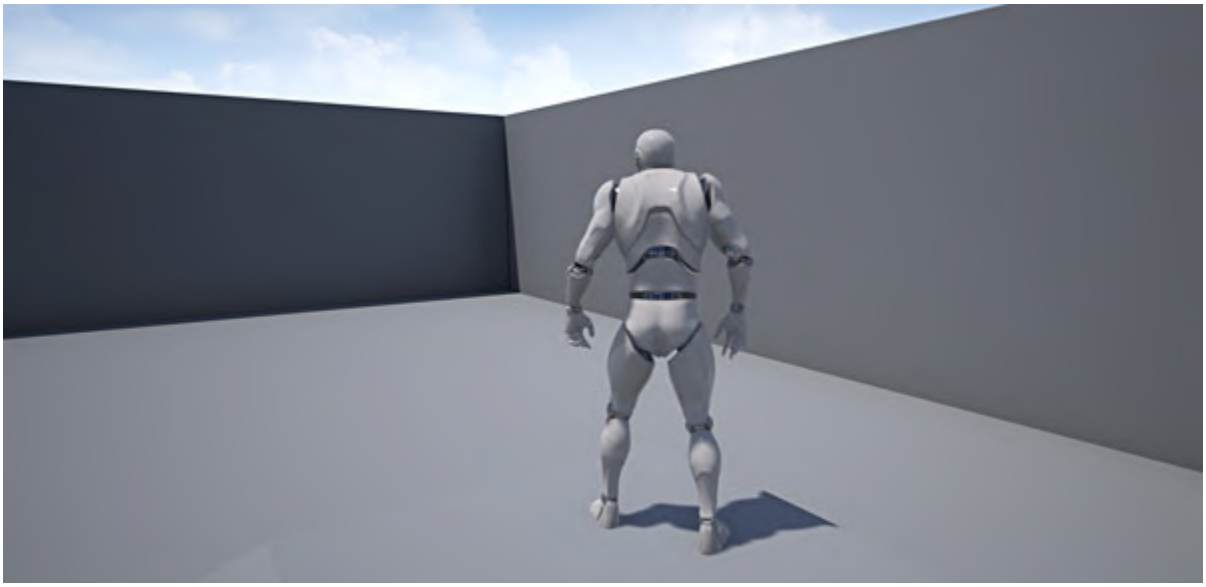
To start with, use Right Click and Play From Here as well as Play icon options.

Depending on which template being used for the project, how you spawn inside the level will vary.

Spawning inside the level with FPS Shooter Template:



Spawning inside the level with Third Person Template:



26. BUILD ALL

Last option you will need to know as a beginner is how to build your map.

Build function renders and bakes lighting information, builds geometry, generates lightmaps and shadow maps, builds level of detail and generates navigation paths.

You will find the Build icon within the top toolbar:

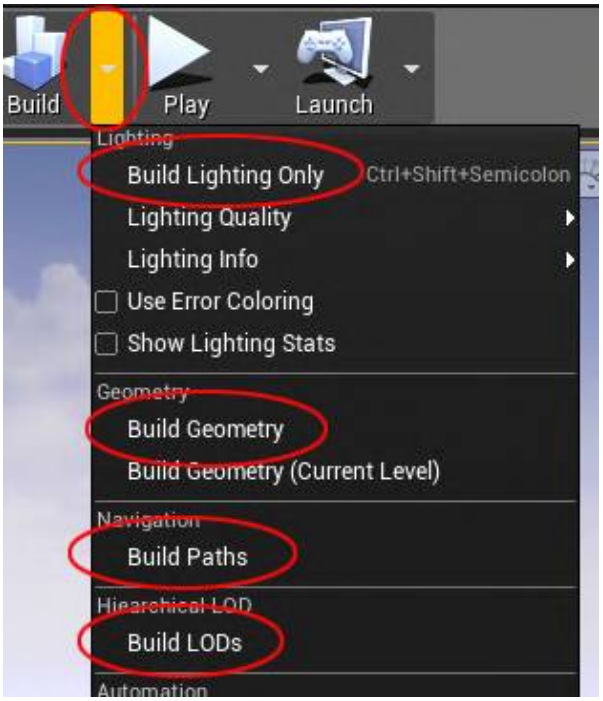
What you currently see inside the perspective viewport is not what the level actually looks like. It is close, but not final. You have to Build to see final lighting and geometry results.

There are 4 quality options to choose from prior to Build:

- Preview
- Medium
- High
- Production

Preview is used for quick builds and during most of the level construction. Switch to Production when you are ready to finalize your level. Production Build will take the longest and will produce the best visual quality result.

And last, you can choose what aspect of your level you want to Build. Use a drop down menu and select:

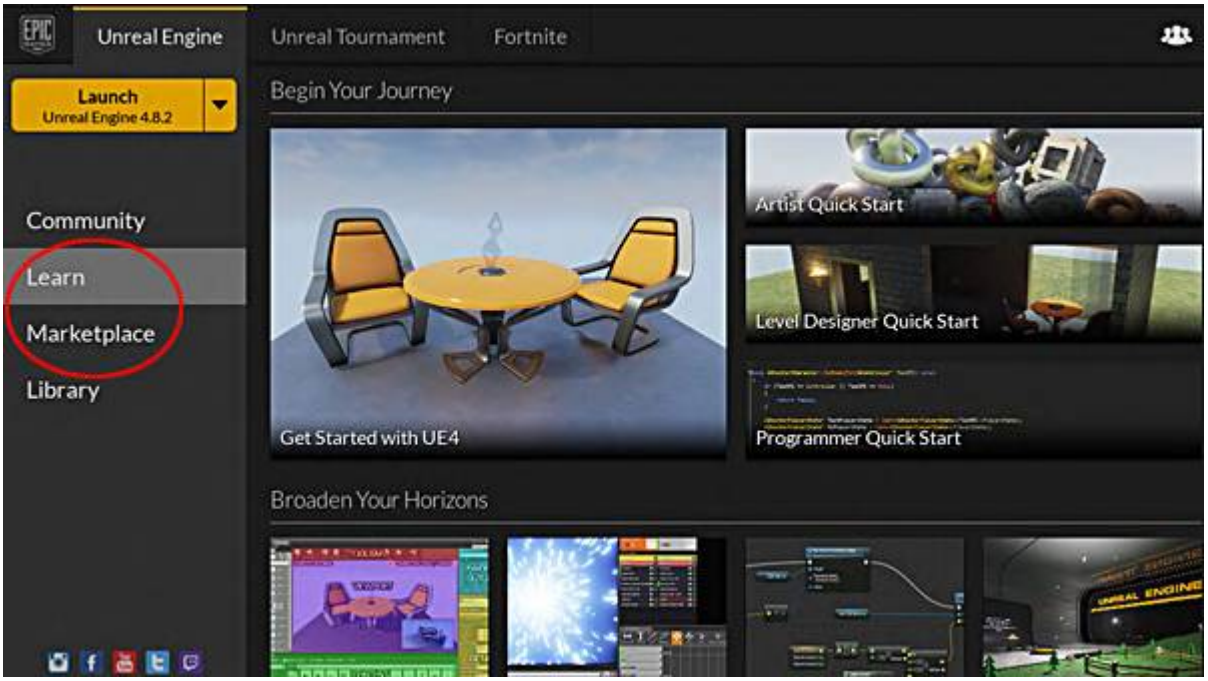


- Build Geometry
- Build Lighting
- Build Paths
- Build LOD
- Build All (will build all of the above together)

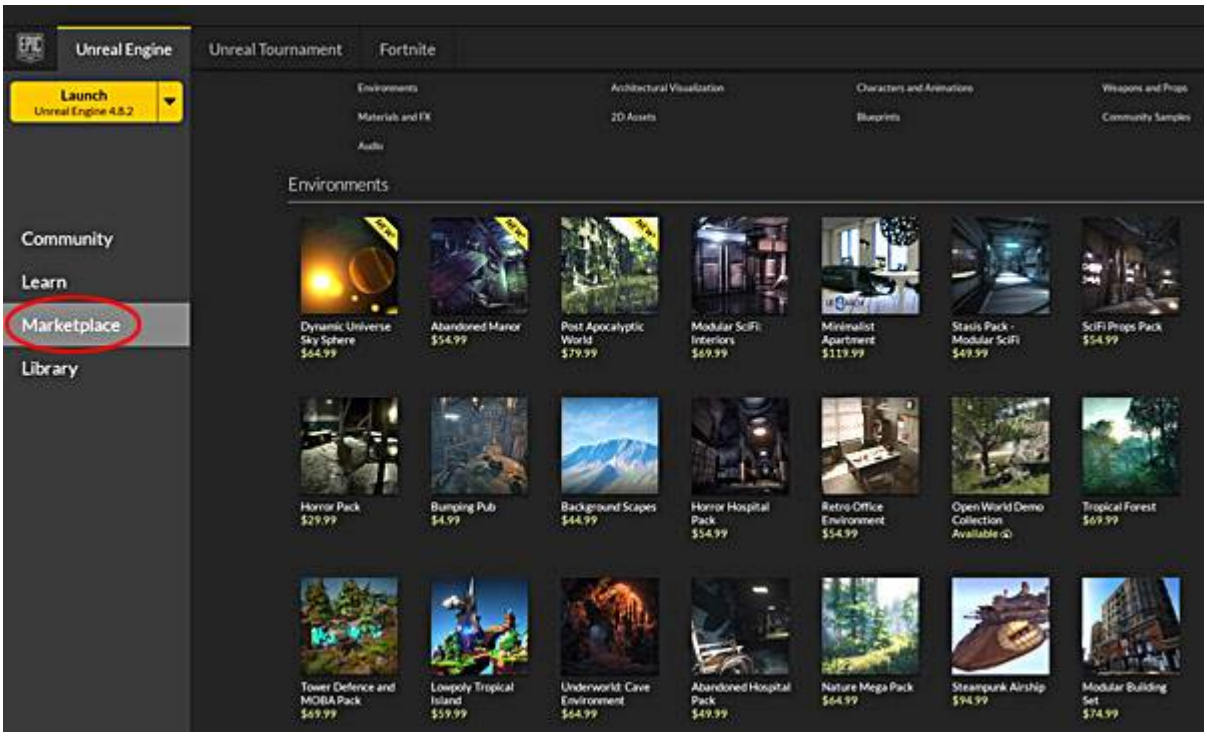
You would start with Building on Preview when you need to see updated and quick results of your lighting. This allows continuing working without a lot of waiting. Once you begin to near the end of the level creation or you need to see how your environment looks with higher visual quality - switch to High or Production. Final Build All should always be set to Production.

27. MARKETPLACE CONTENT AND LEARN SECTION

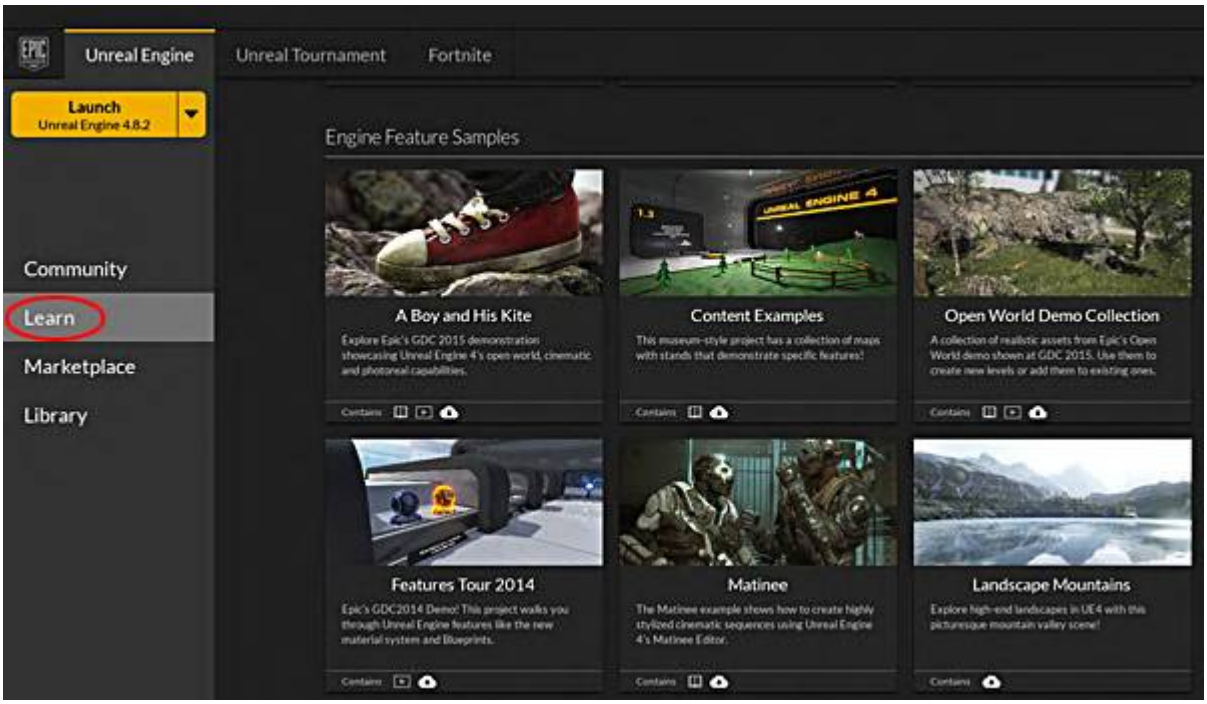
Marketplace and Learn sections can be found inside the Unreal Engine Launcher:



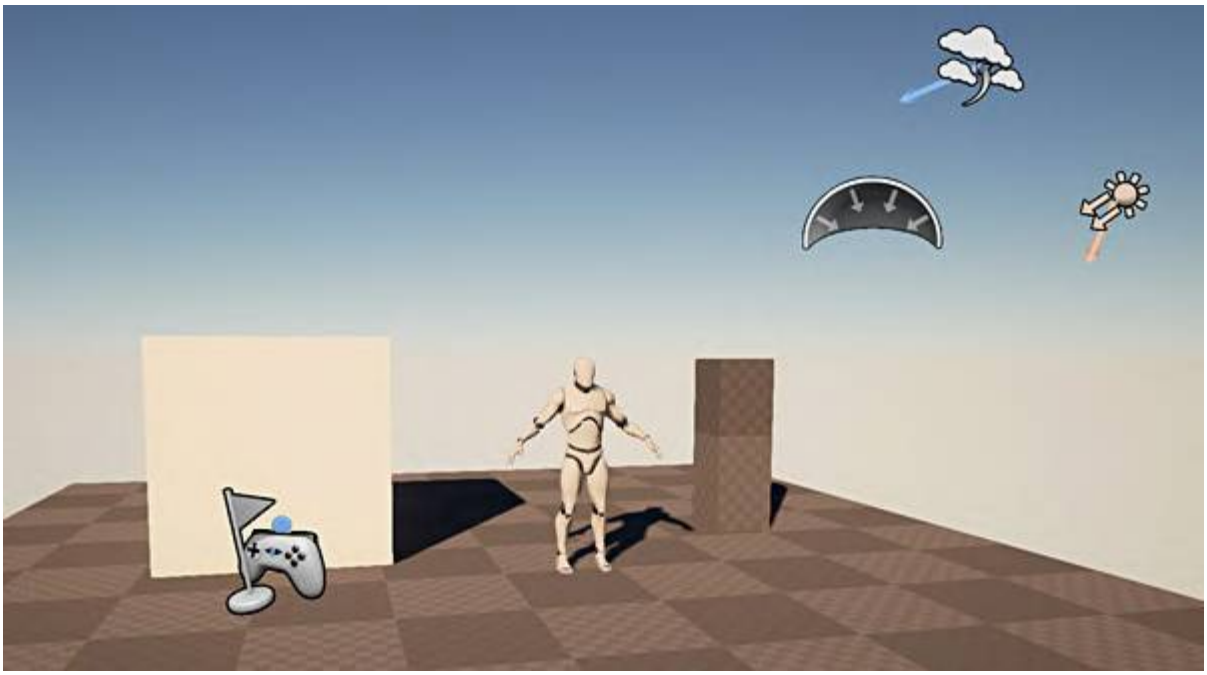
Through Marketplace you can purchase and download Static Meshes, characters, animations, sounds, particle effects etc. All of these can be used to help you construct game environments without you having to create these assets on your own:



Through Learn section you can download free game engine samples and game examples. You can use these to reverse engineer how they were created to help you understand what you would need to do yourself:


Unreal Engine
Unreal Tournament
Fortnite
Launch
Unreal Engine 4.8.2
Engine Feature Samples
Community
Learn
Marketplace
Library
A Boy and His Kite
Explore Epic's GDC 2015 demonstration showcasing Unreal Engine 4's open world, cinematic and photoreal capabilities.
Contains
Content Examples
This museum-style project has a collection of maps with stands that demonstrate specific features!
Contains
Open World Demo Collection
A collection of realistic assets from Epic's Open World demo shown at GDC 2015. Use them to create new levels or add them to existing ones.
Contains
Features Tour 2014
Epic's GDC2014 Demo! This project walks you through Unreal Engine features like the new material system and Blueprints.
Contains
Matinee
The Matinee example shows how to create highly stylized cinematic sequences using Unreal Engine 4's Matinee Editor.
Contains
Landscape Mountains
Explore high-end landscapes in UE4 with this picturesque mountain valley scene!
Contains

SECTION 3: BEGINNER'S STEP-BY-STEP TO CREATING YOUR FIRST PLAYABLE LEVEL



How do you create your first playable level inside Unreal Engine 4?

As a beginner, you want to get started with a game engine/level editor as quickly as possible and build something you can play in. For this you need to know what objects or actors (as they are called in UE4) that are required to be in your level.

In this section you will learn how to create a starter level that contains:

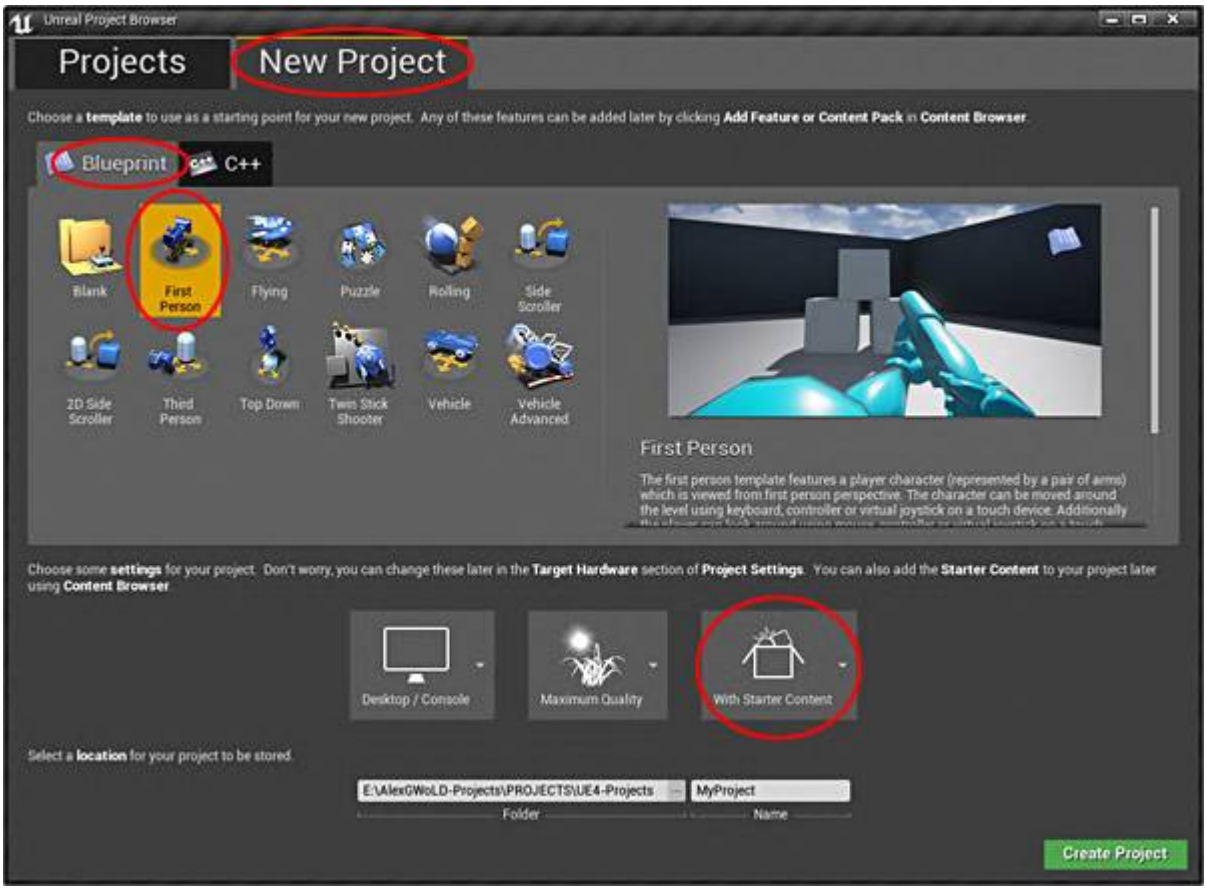
- Ground plane
- Player start
- Player scale reference to judge proportion
- Fog
- Sunlight
- Static Mesh (3d model)
- Build and Play Test your level

Let's get started.

SETTING UP

You should have a project created using any of the available game templates and with Starter Content.

I will be using Blueprint FPS Shooter game template and Starter Content:

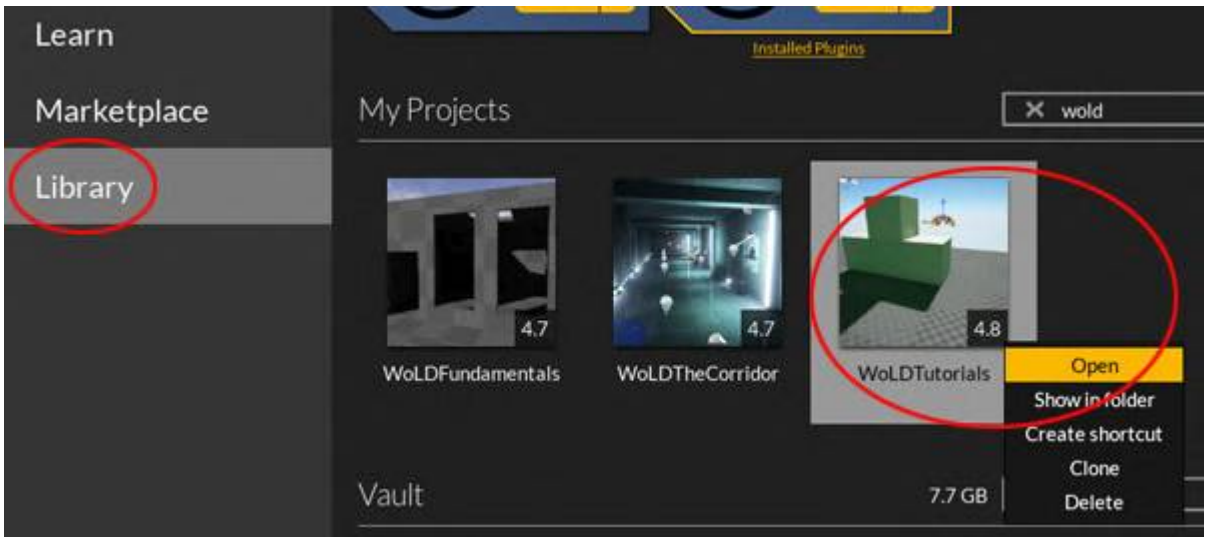

Unreal Project Browser
Projects
New Project
Choose a template to use as a starting point for your new project. Any of these features can be added later by clicking Add Feature or Content Pack in Content Browser.
Blueprint
C++
Blank
First Person
Flying
Puzzle
Rolling
Side Scroller
2D Side Scroller
Third Person
Top Down
Twin Stick Shooter
Vehicle
Vehicle Advanced
First Person
The first person template features a player character (represented by a pair of arms) which is viewed from first person perspective. The character can be moved around the level using keyboard, controller or virtual joystick on a touch device. Additionally
Choose some settings for your project. Don't worry, you can change these later in the Target Hardware section of Project Settings. You can also add the Starter Content to your project later using Content Browser.
Desktop / Console
Maximum Quality
With Starter Content
Select a location for your project to be stored.
E:\AlexGWoLD-Projects\PROJECTS\UE4-Projects
MyProject
Folder
Name
Create Project

If you don't have a project set up, follow this tutorial. It will explain everything you need to do to start your own project and launch the editor.

LAUNCH THE EDITOR

Open your project to follow along.

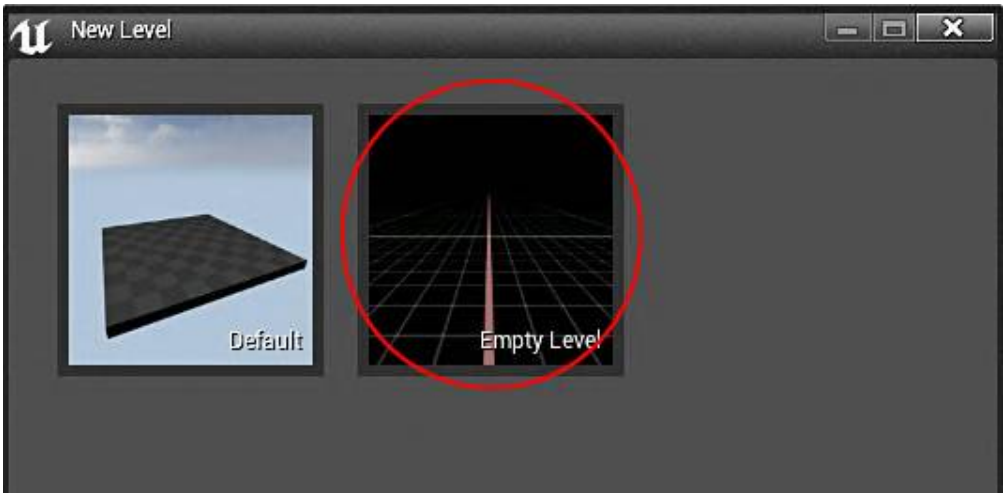
Inside the Epic Games Launcher switch over to Library tab and double click on your project image thumbnail or right click and choose open to launch the editor:



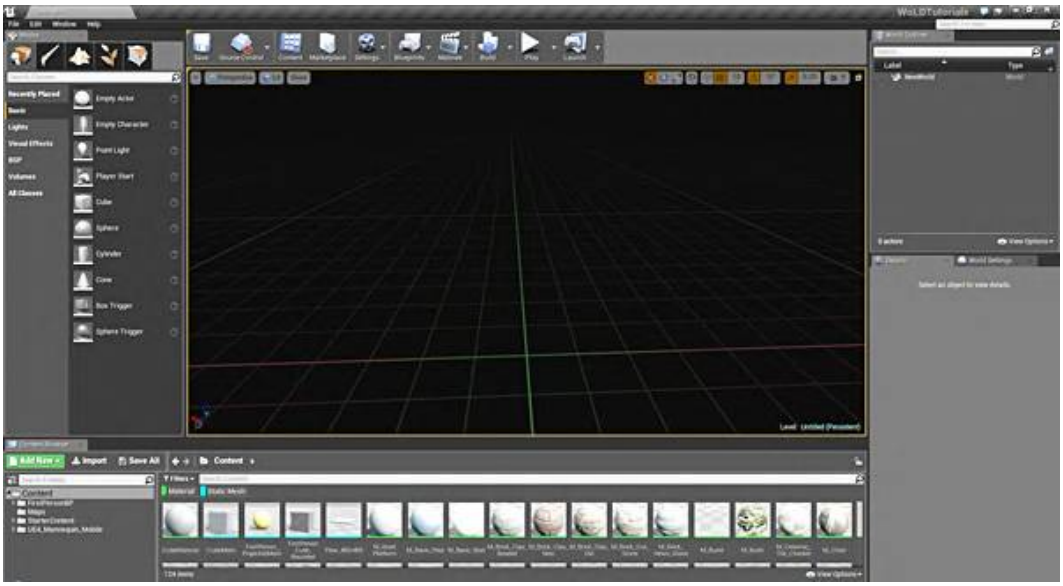
START A NEW BLANK LEVEL

With the editor open, create an empty level. Go to File > New Level:

Choose Empty Level:



We are going to start with a blank void map and we'll insert all of the needed actors into it.

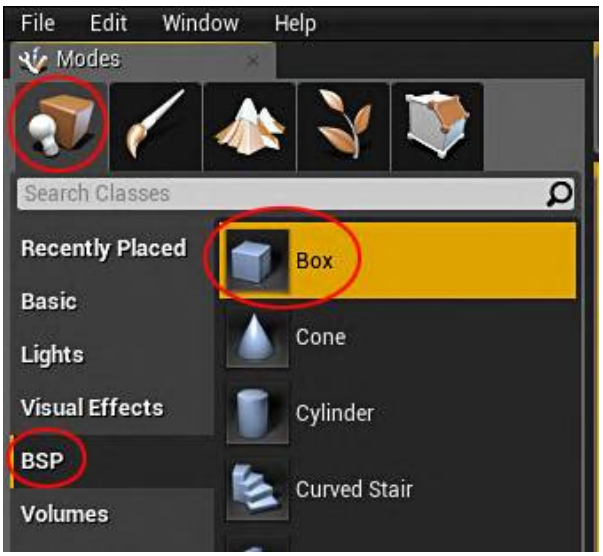


GROUND PLANE

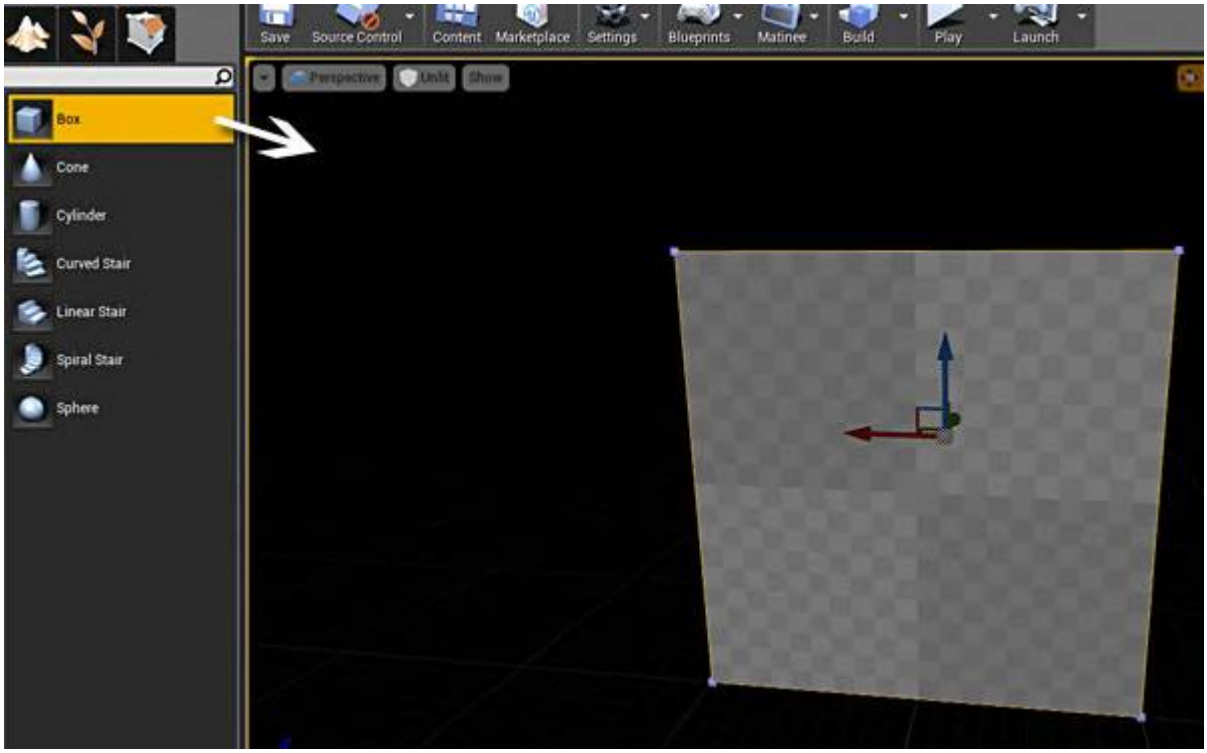
We need a ground plane, a floor onto which the player can stand on.
For ground plane we can use:

- BSP Brush or Static Mesh (3d model)

We'll use a brush. Go to Place Modes and switch to BSP:



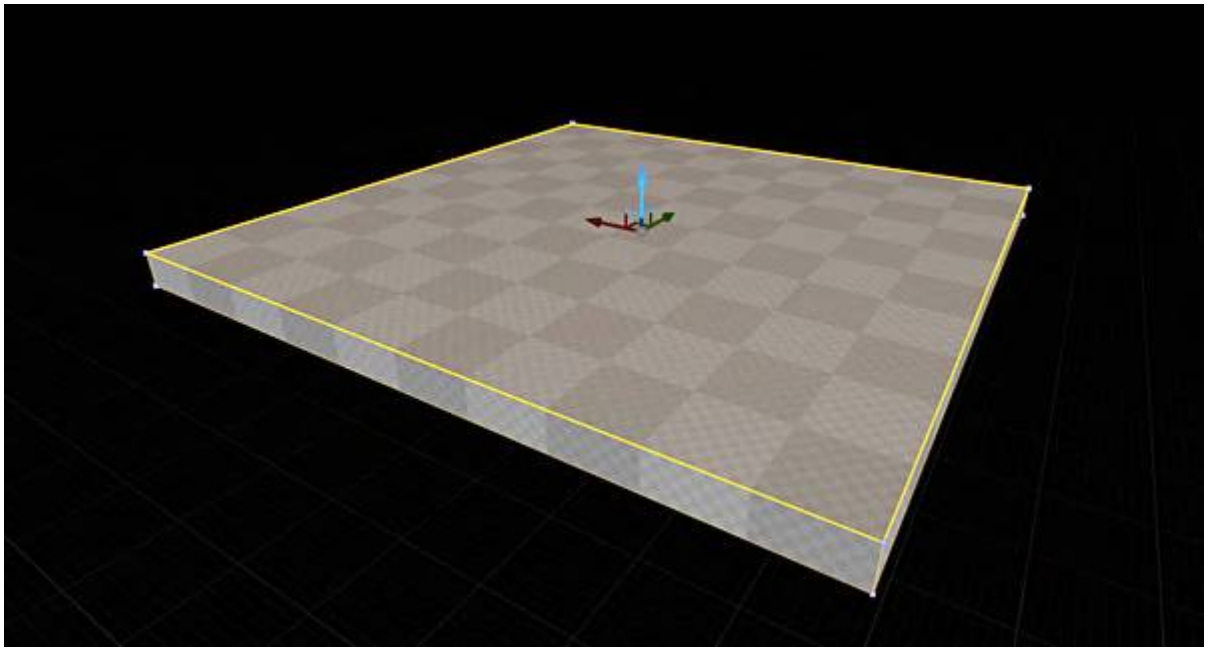
Left click and drag a BSP Box into perspective viewport:



With the BSP brush selected, go to Details panel and resize the brush to following values:

- X = 1000
- Y = 1000
- Z = 50

We now have a ground plane:



CHARACTER REFERENCE SCALE

You want to build your worlds to correct scale and proportion. No object or architecture should be too large or too small otherwise it will ruin the illusion of your environment very quickly.

To do this we'll insert a character reference with the same dimensions as a player model in game. It will be used to judge scale as we build the level. Since we aren't creating this level for any specific game that contains a specific collision model for the player character, we'll use UE4 base dimensions.

Base Character Dimensions in UE4 are:

- Height = 180 cm/uu
- Width/Depth = 60 cm/uu

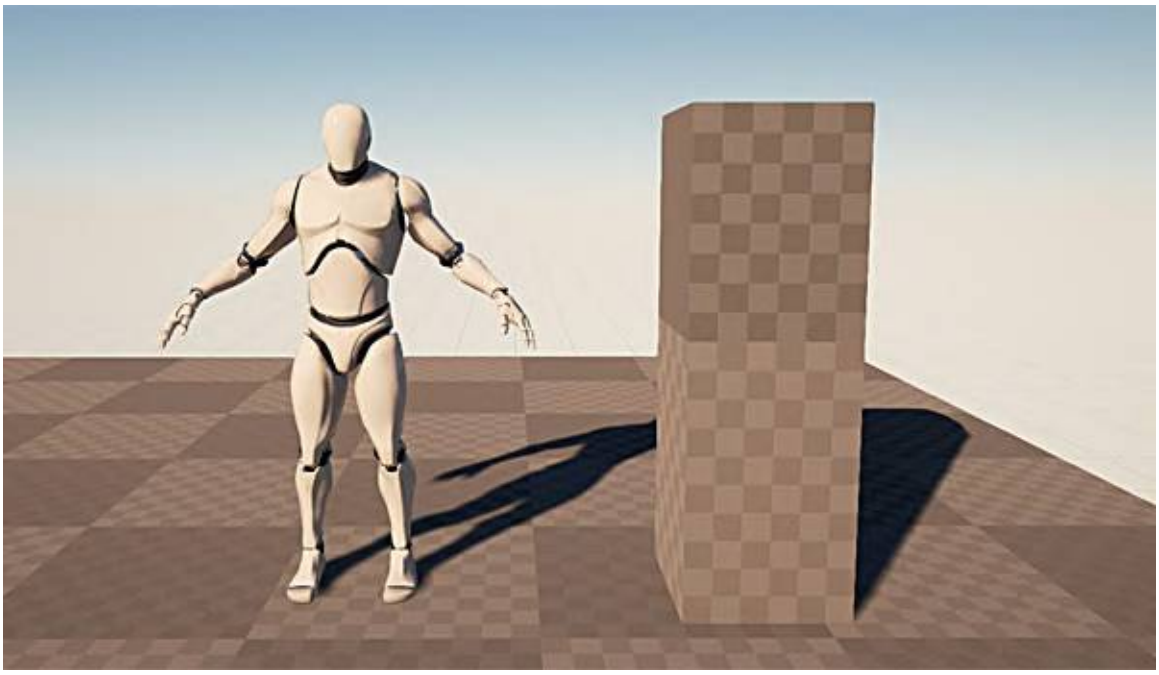
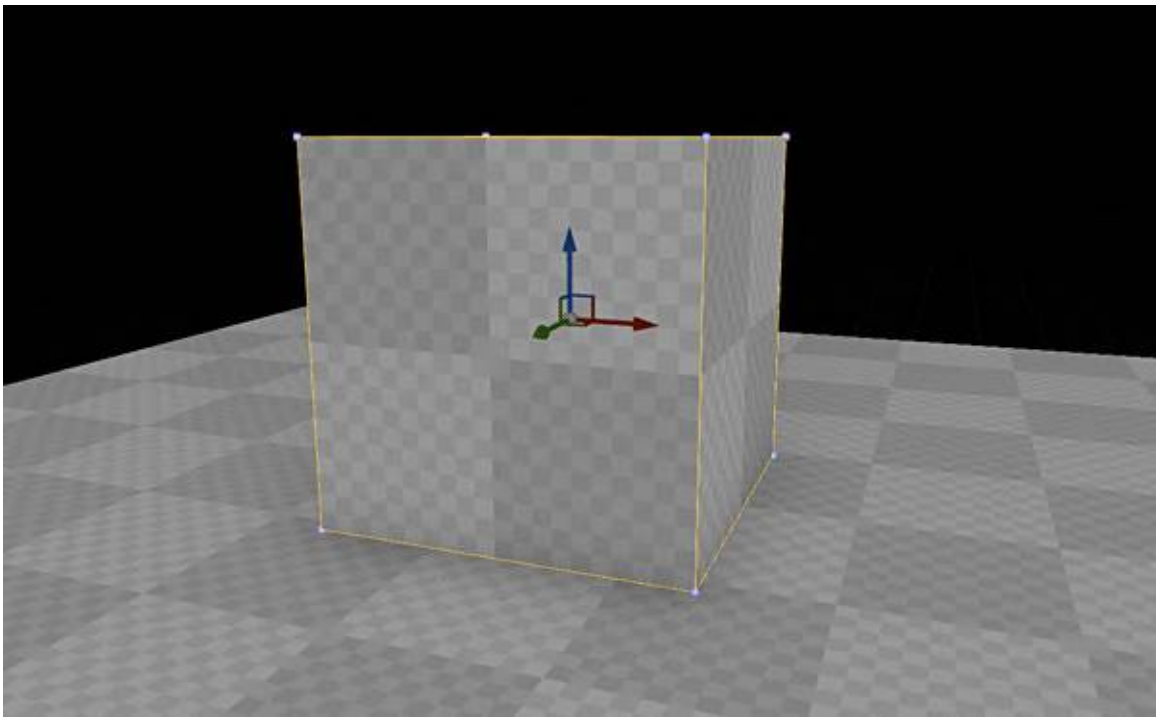


Image above is just a comparison between a character mesh and BSP brush with same dimensions.

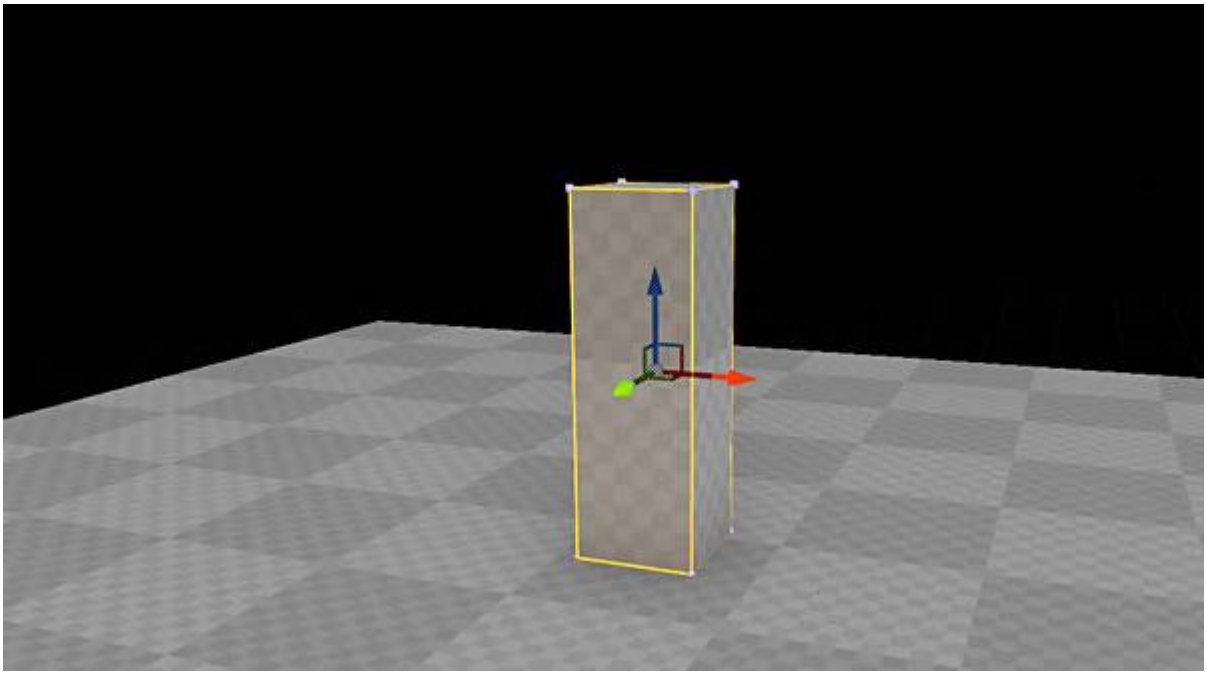
Let's use a BSP brush for this.

Drag a BSP box from the Place Modes panel into the perspective viewport and onto the ground plane:



In the Details panel enter the following values to resize:

- X = 60
- Y = 60
- Z = 180



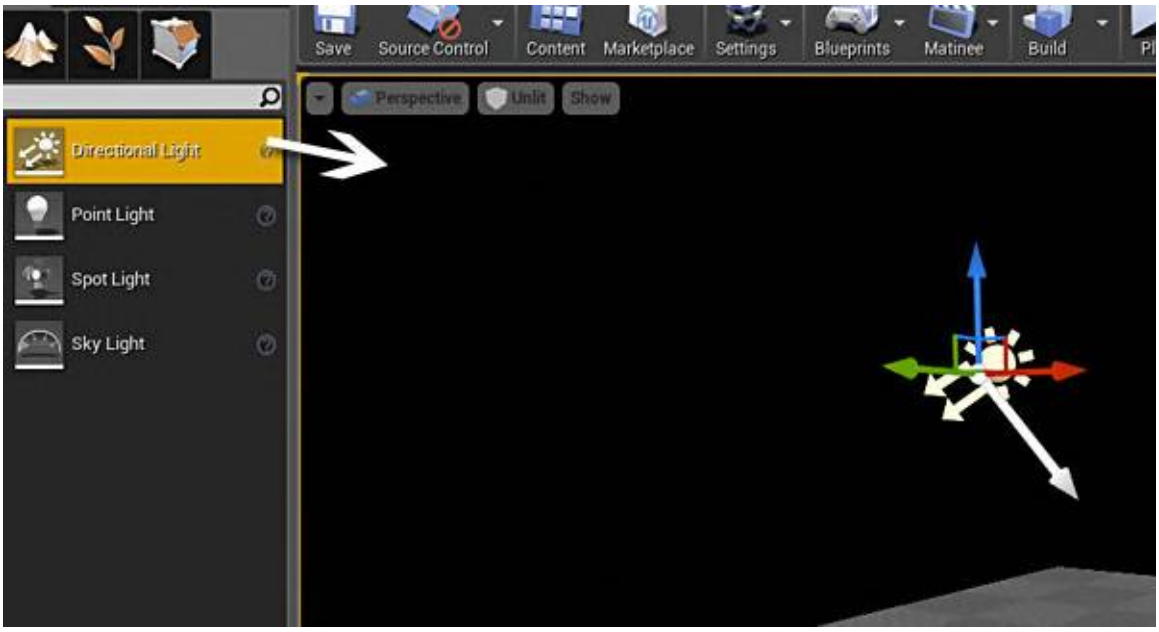
For character reference you can use other options. Such as a Static Mesh box, Skeletal Mesh Reference provided with UE4 or your own custom mesh.

DIRECTIONAL LIGHT

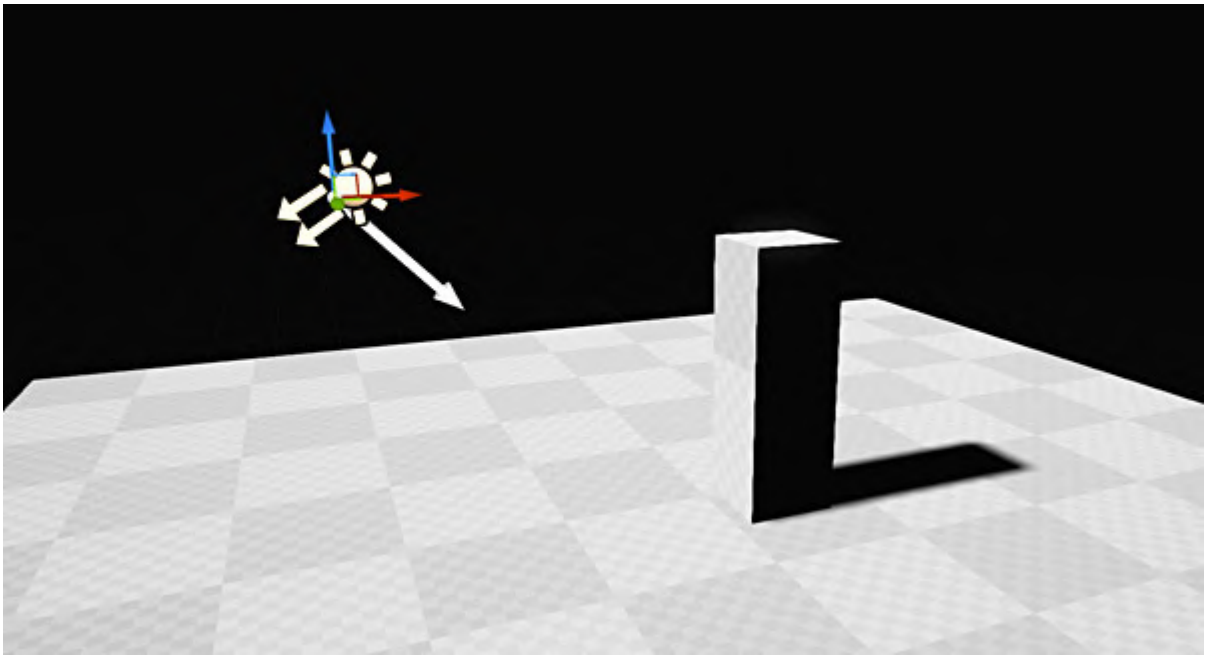
Now, we need to insert a Directional Light; this is our sun light actor that will illuminate our entire environment.

Go to Place Modes (Shift+1) and switch to Lights tab:

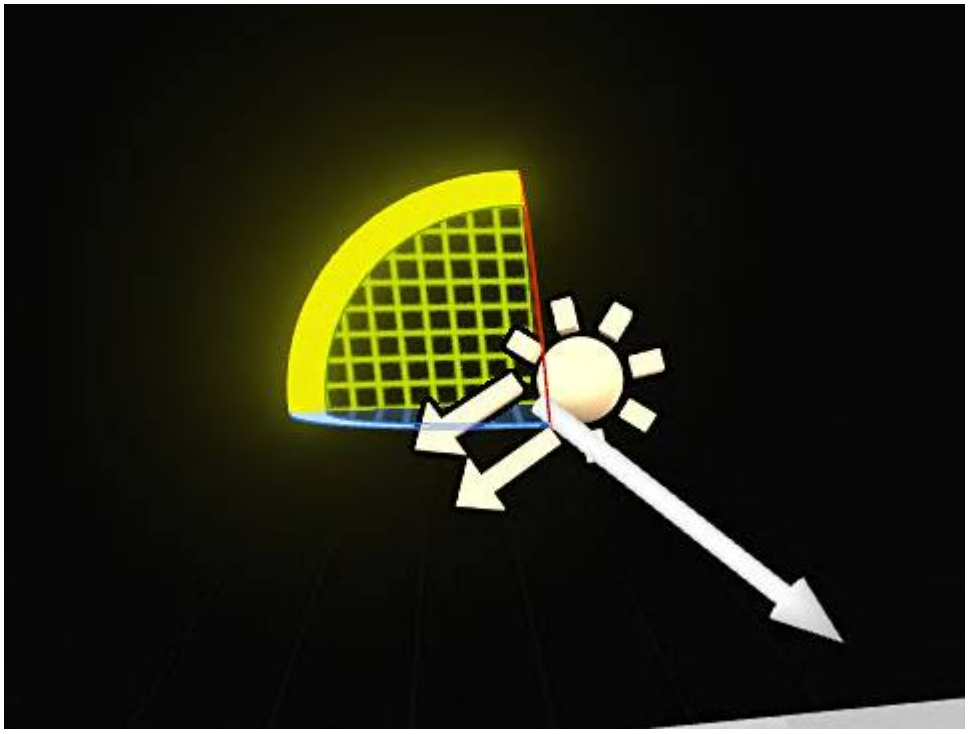
Left click and drag Directional Light actor into perspective viewport. You will see Directional Light take effect of lighting your world:



If you do not, switch over to Lit Mode (Alt+4):

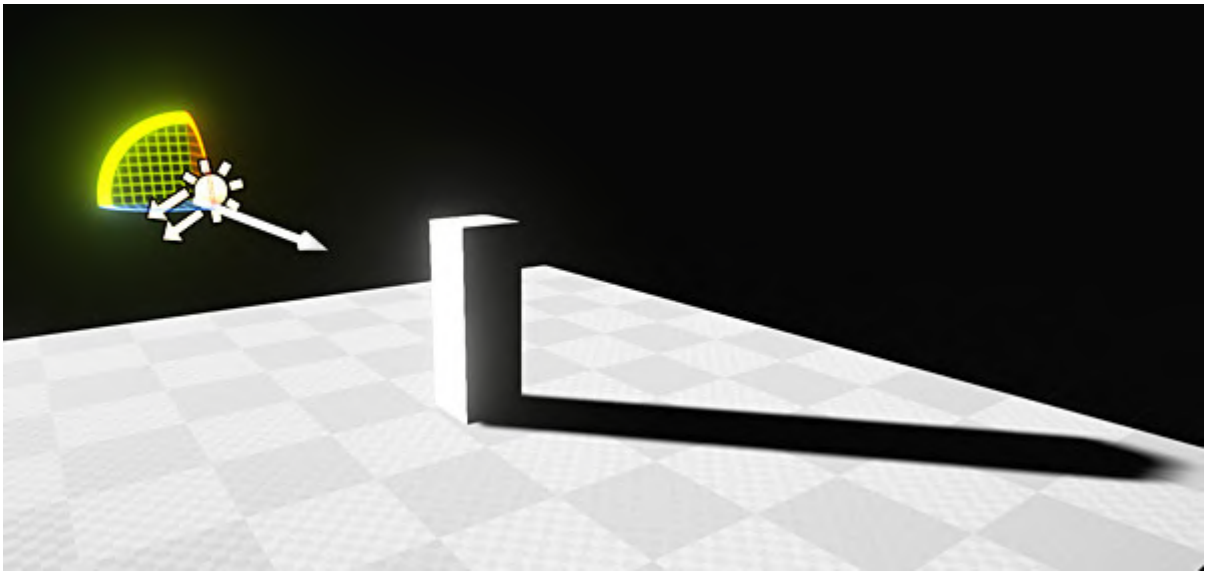


Select the Directional Light and use the rotate gizmo (E) or hit spacebar until you see rotation handles. Rotate and point the light towards the ground plane for angle and direction you want for the environment.

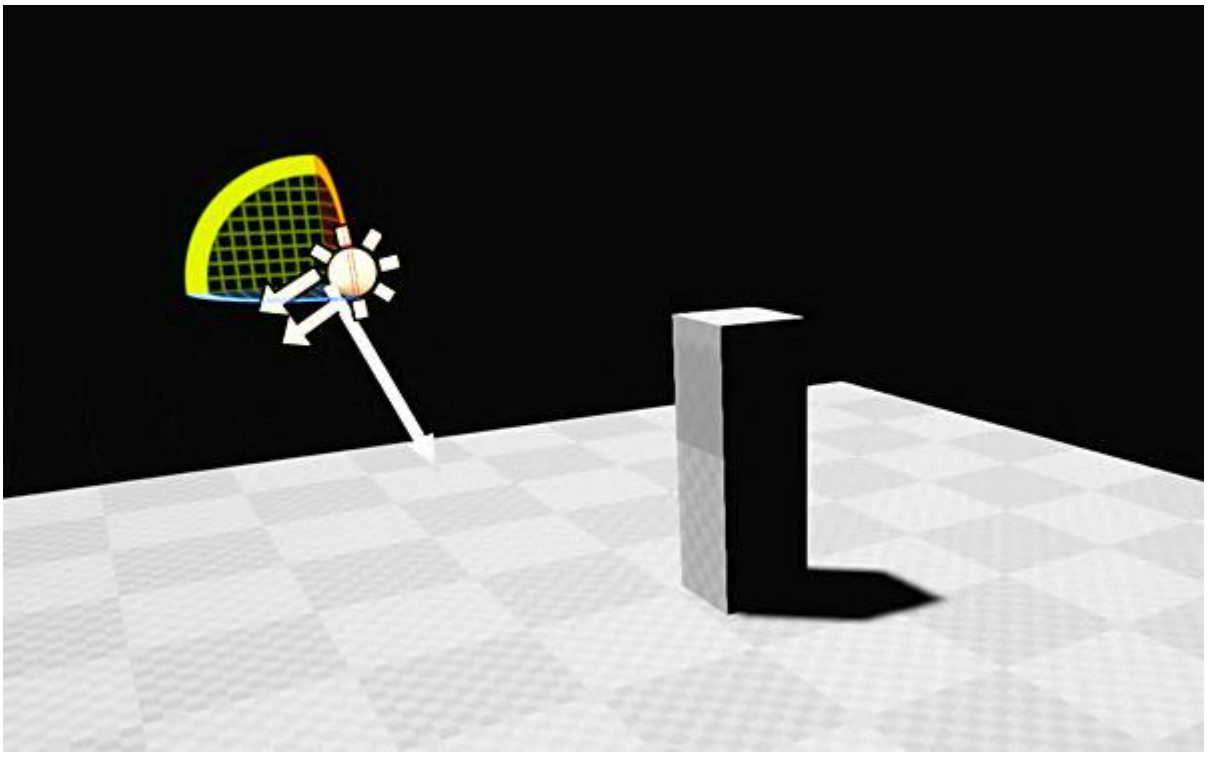


For example, for early morning or late evening, aim the Directional Light angle very low to cast long shadows across your level. For afternoon, point the light down.

Morning/Evening Angle:



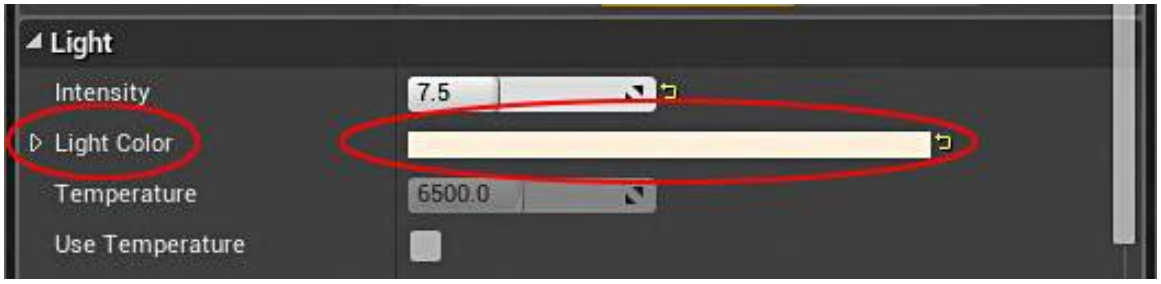
Afternoon Angle:



First two properties to change are color and intensity.

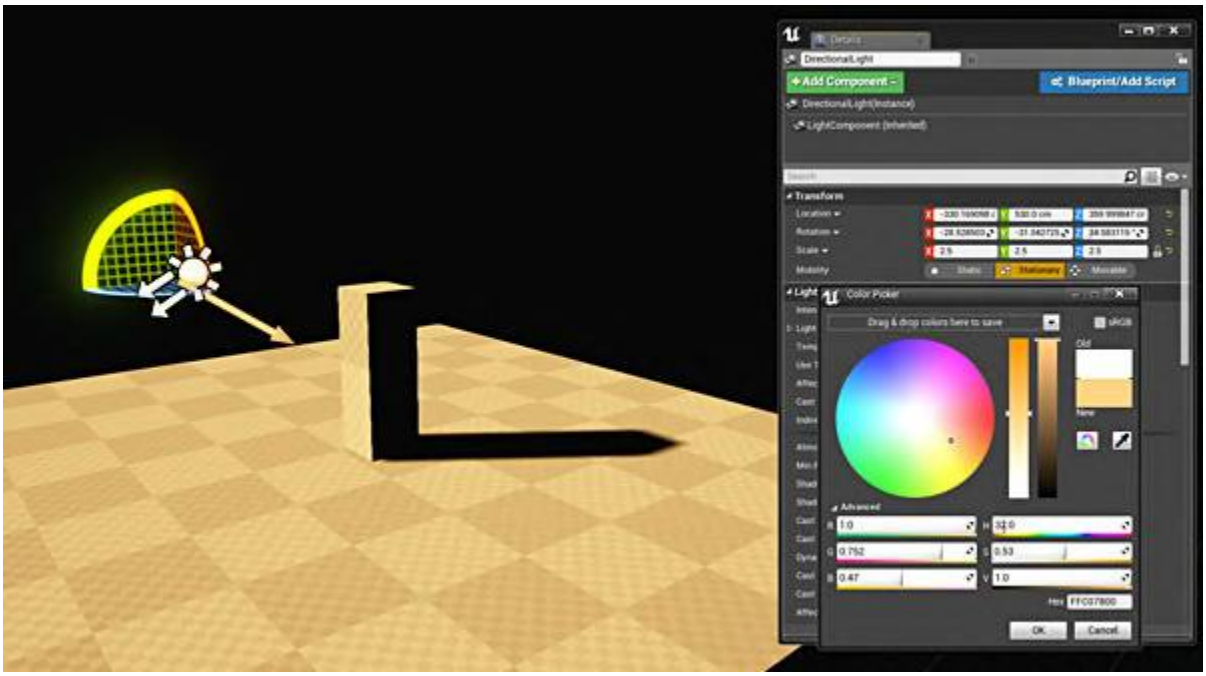
- Color: sets the color of your light
- Intensity: strength of the light

Select the Directional Light and in the Details panel left click on the color bar under Color:

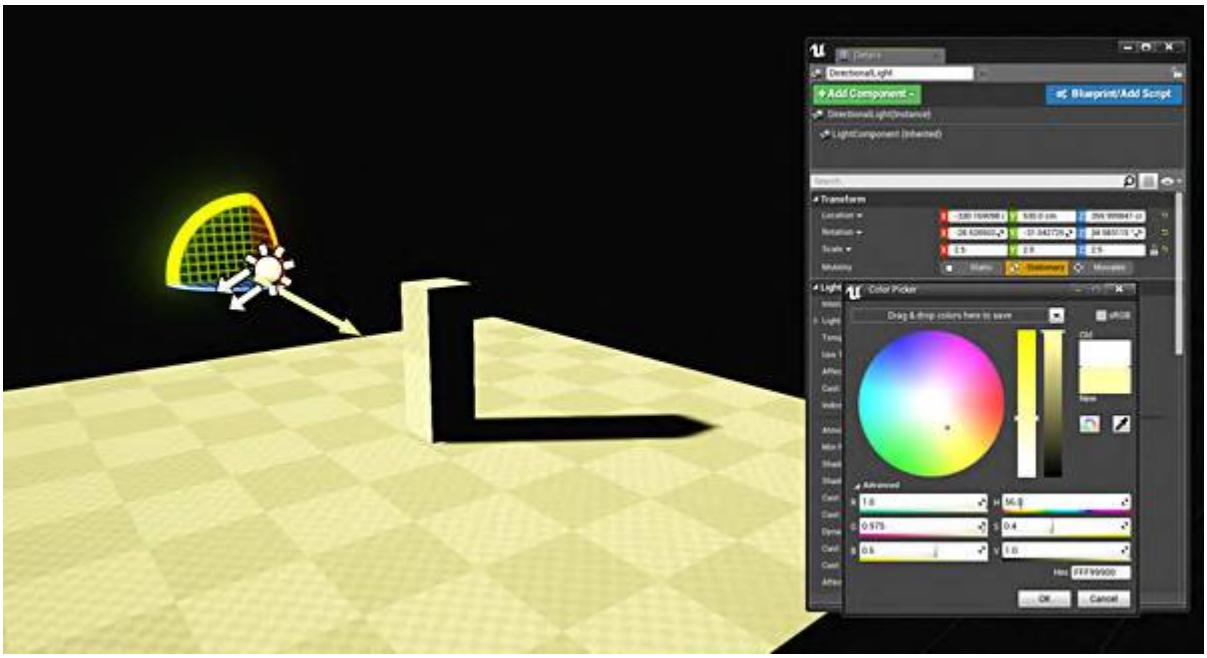


Use the Color Picker to choose the color you want:

Color you choose should reflect the time of day you want. Late evening most likely have deeper orange or reds:

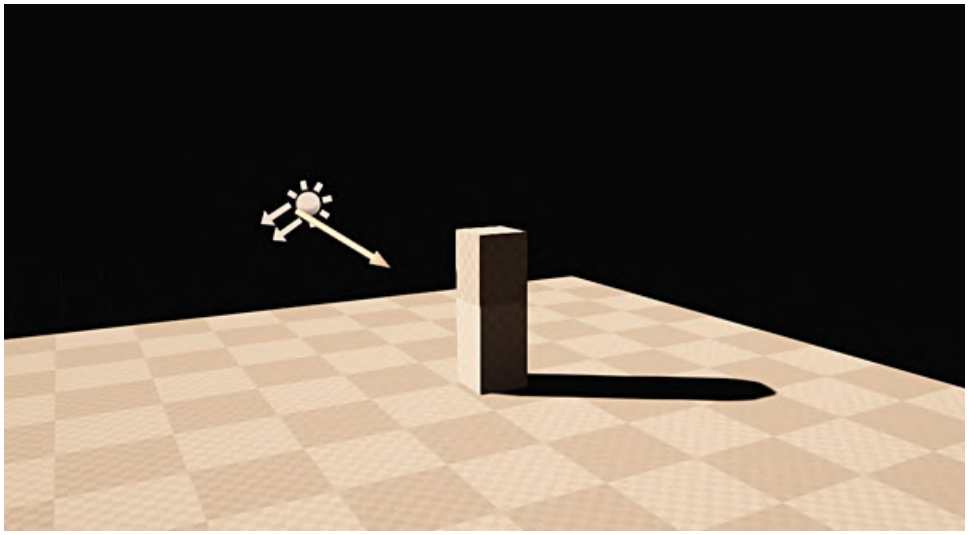


While early morning might be more on the lighter orange and yellows:



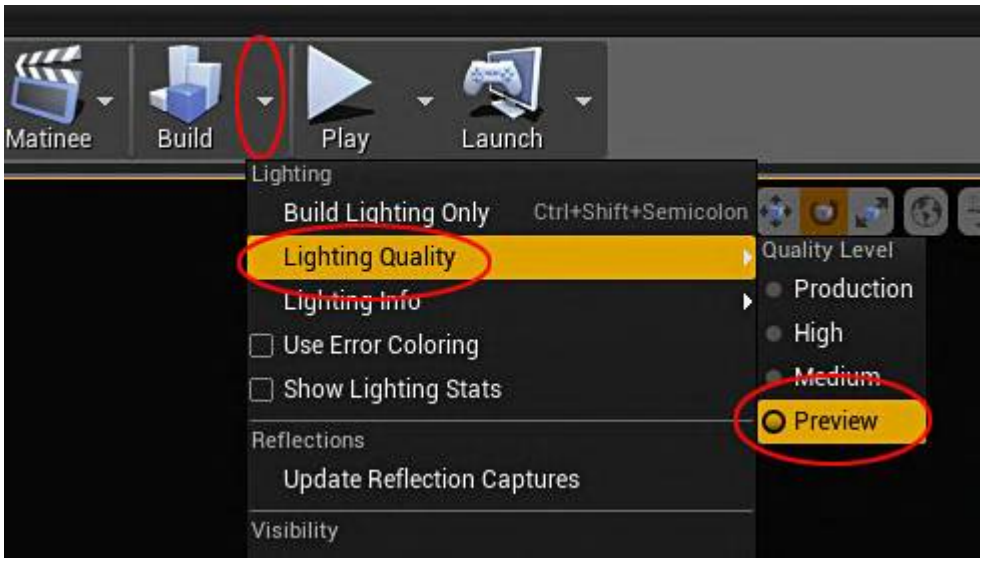
Once you've chosen the color, click OK.

I chose lighter orange/peach color:



Let's set Intensity; this is the strength of the Directional Light. Value of 10 is a bit too high so let's bump this down to 7.5:

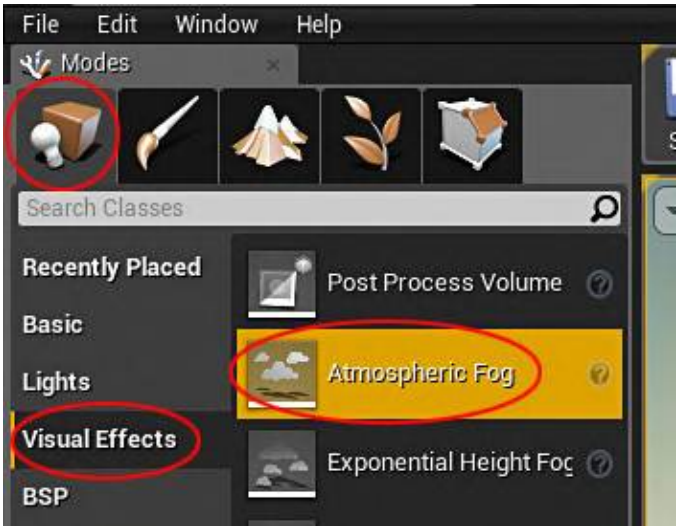
To see accurate light representation, we need to Build our level. Use the drop down Build option and set Lighting Quality to Preview:



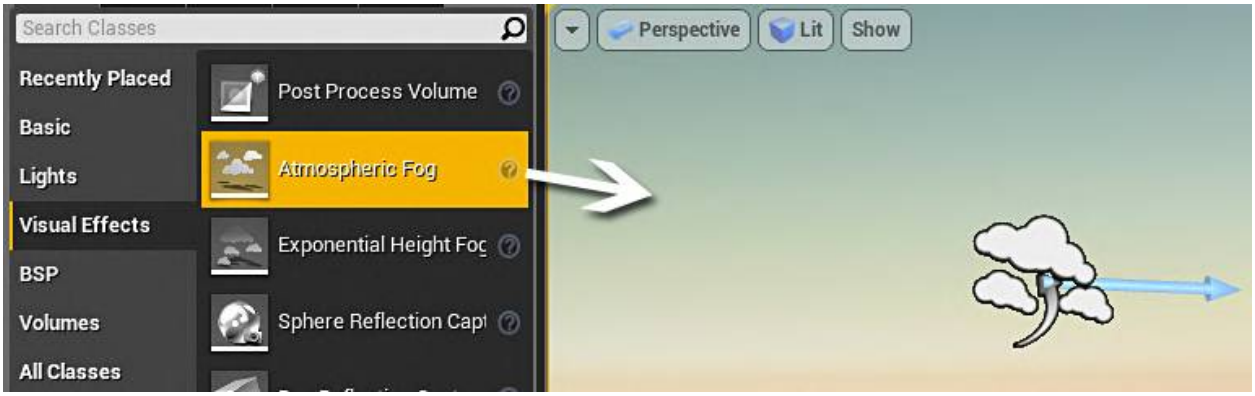
Preview will allow us to continue working without waiting too long. Eventually, towards the end of your level creation, you will begin building on High and Production Quality for best visual results. Click “Build”:

ATMOSPHERIC FOG

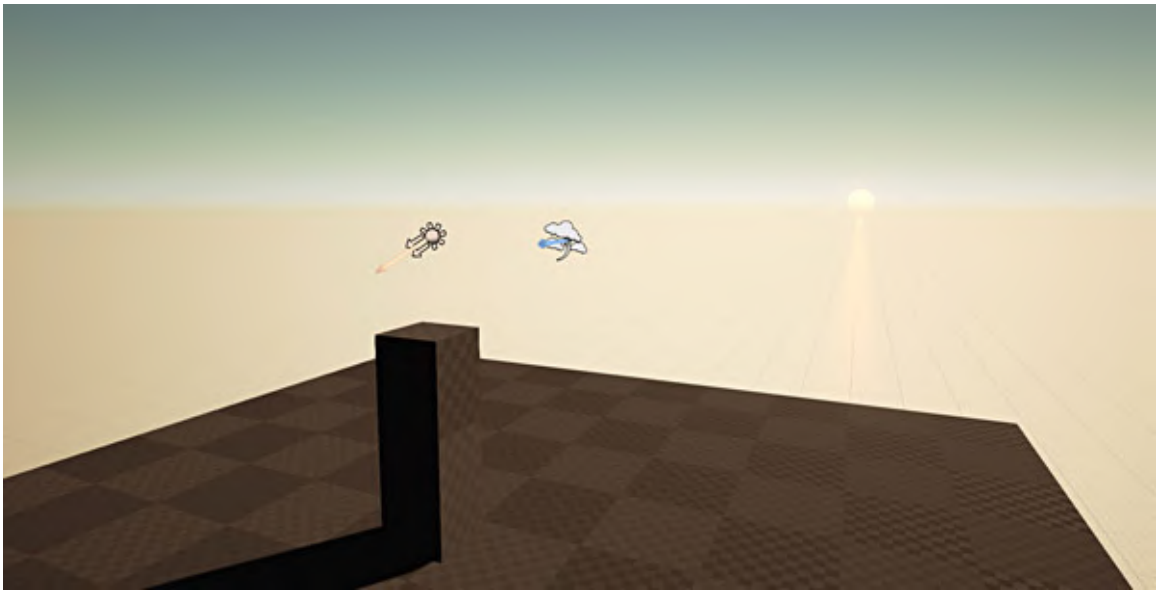
Let's insert Atmospheric Fog; it will give us more realistic atmosphere in the level. Go to Place Modes and Switch to Visual Effects:



Left click and drag Atmospheric Fog right into the perspective viewport:

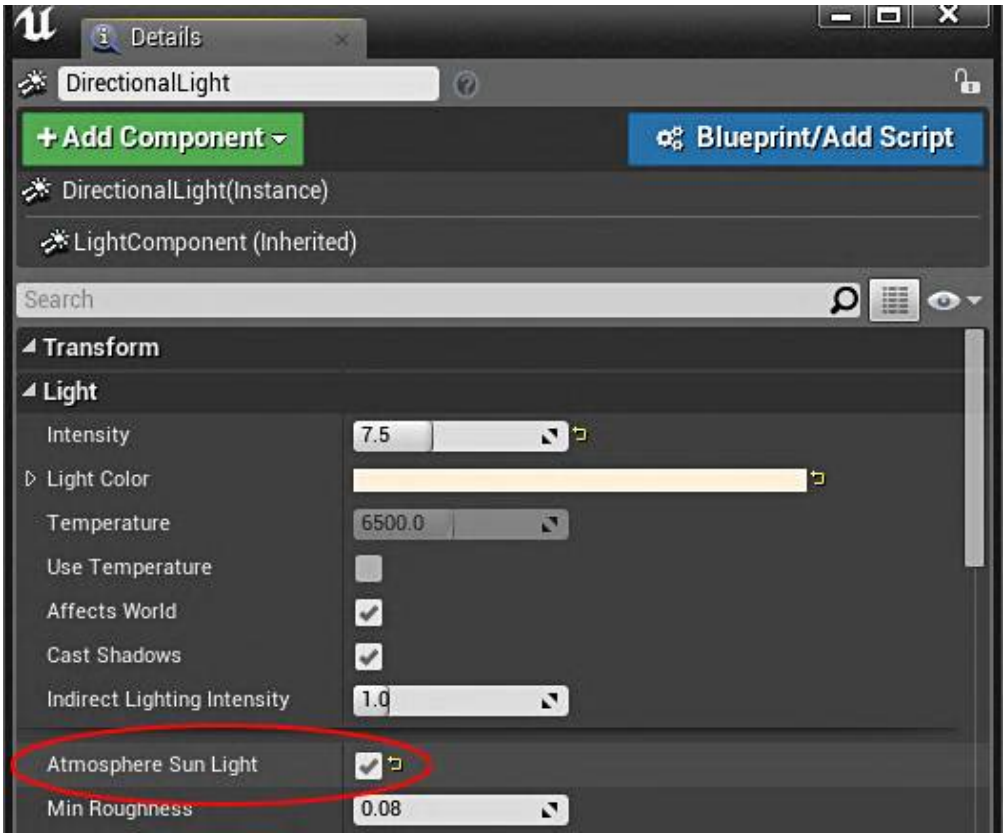


Atmospheric Fog provides an illusion of a skybox, sun disc and distance fog.



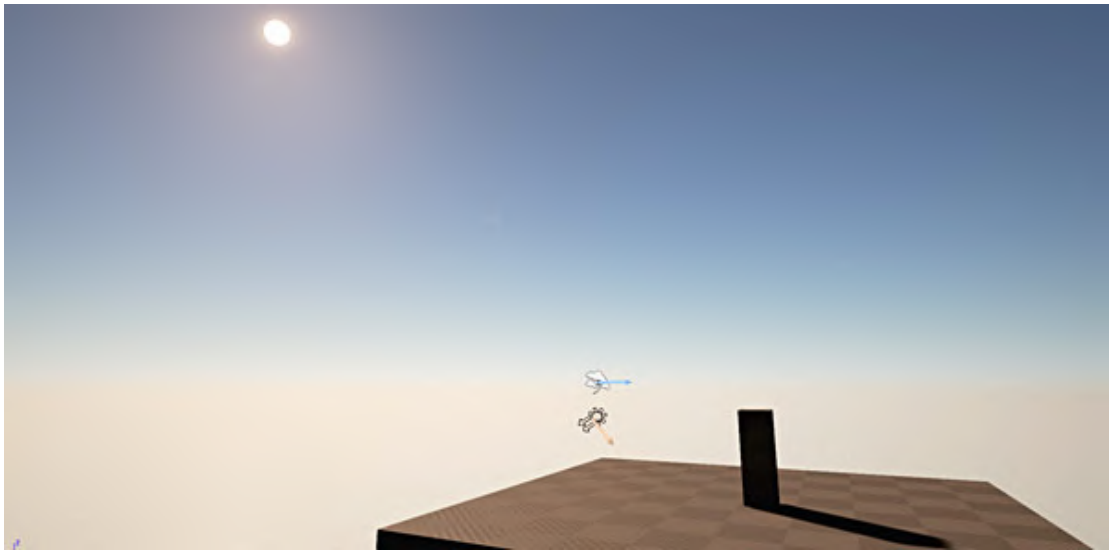
We aren't going to change any properties for Atmospheric Fog but we do need to update settings for Directional Light.

Select Directional Light and Details panel; enable "Atmospheric Sun Light":



This will tie the position and the angle of the sun disc from Atmospheric Fog to match angle and direction of the Directional Light. So any time you want to change the Directional Light angle, atmospheric fog will be updated.

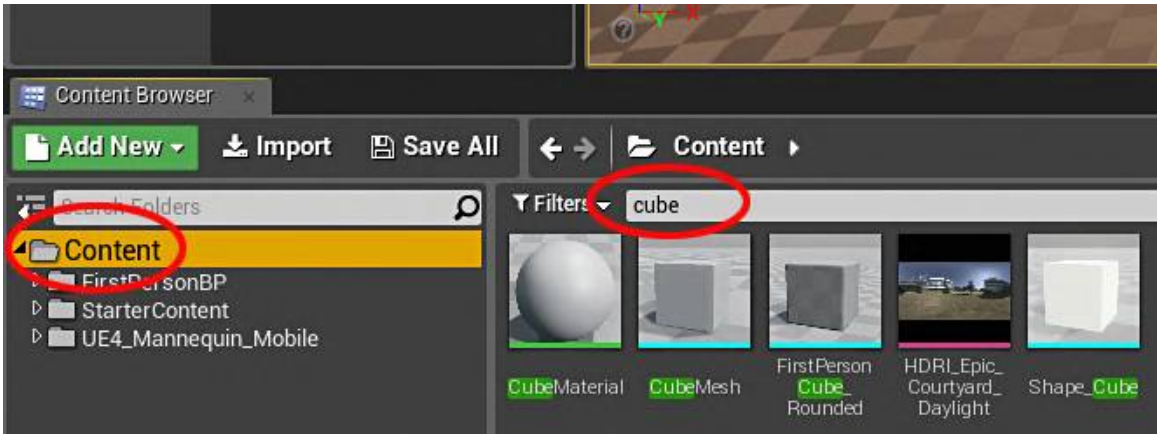
We now have an environment with a light and fog:



INSERT STATIC MESH

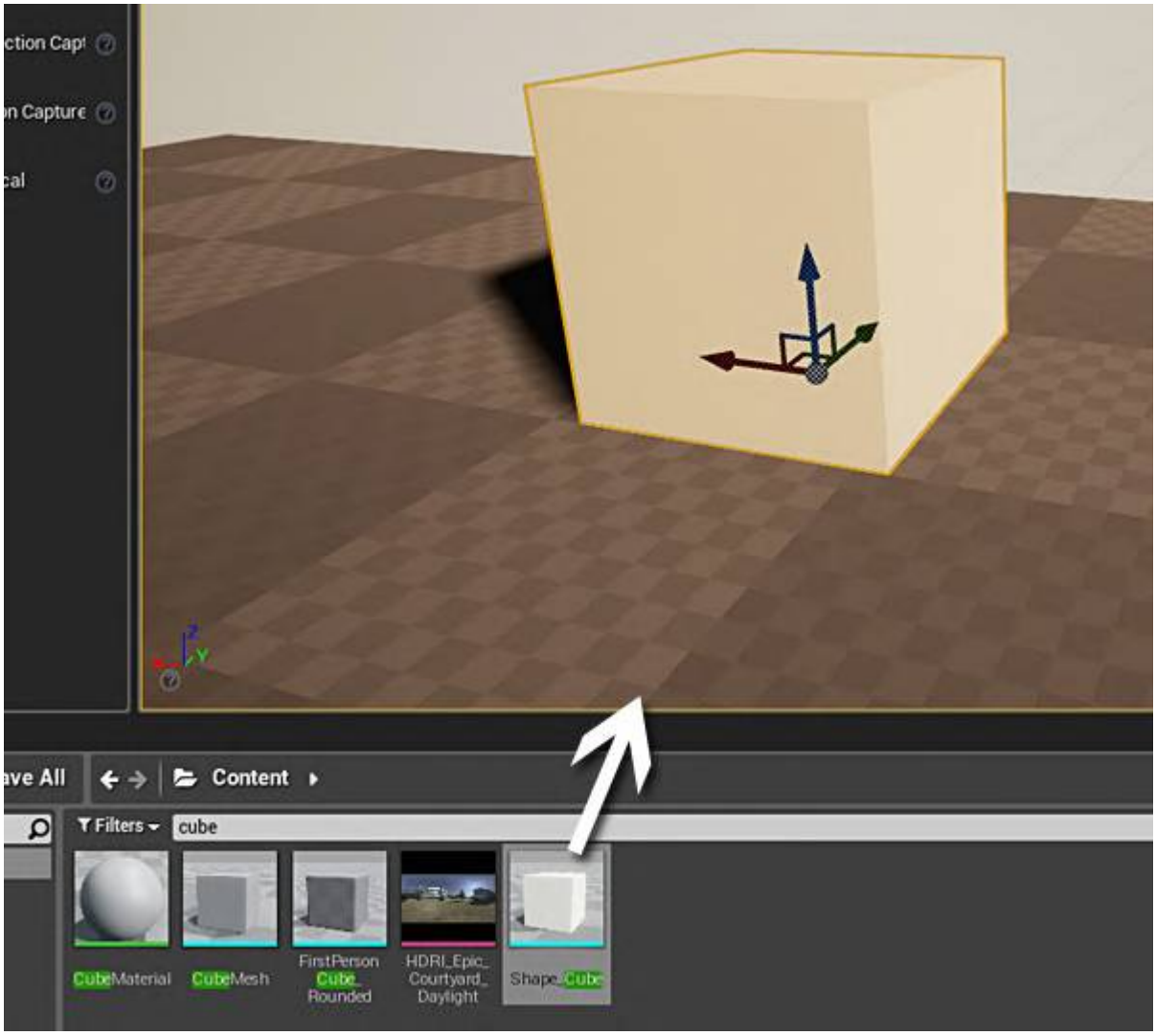
Static Meshes are 3d models created outside Unreal Engine 4, inside a modeling application such as Maya, Maya LT, 3ds Max, Modo, Blender or other and then imported into UE4.

With Starter Content enabled during the project set up, you are given a set of Static Meshes to use. Using Content Browser, search for a Static Mesh. Let's keep it simple and look for Cube:



Make sure to select the "Content" folder so you are searching inside the entire Project directory.

Select the Cube Static Mesh from the Content Browser, left click and drag right into the perspective viewport onto the ground plane:



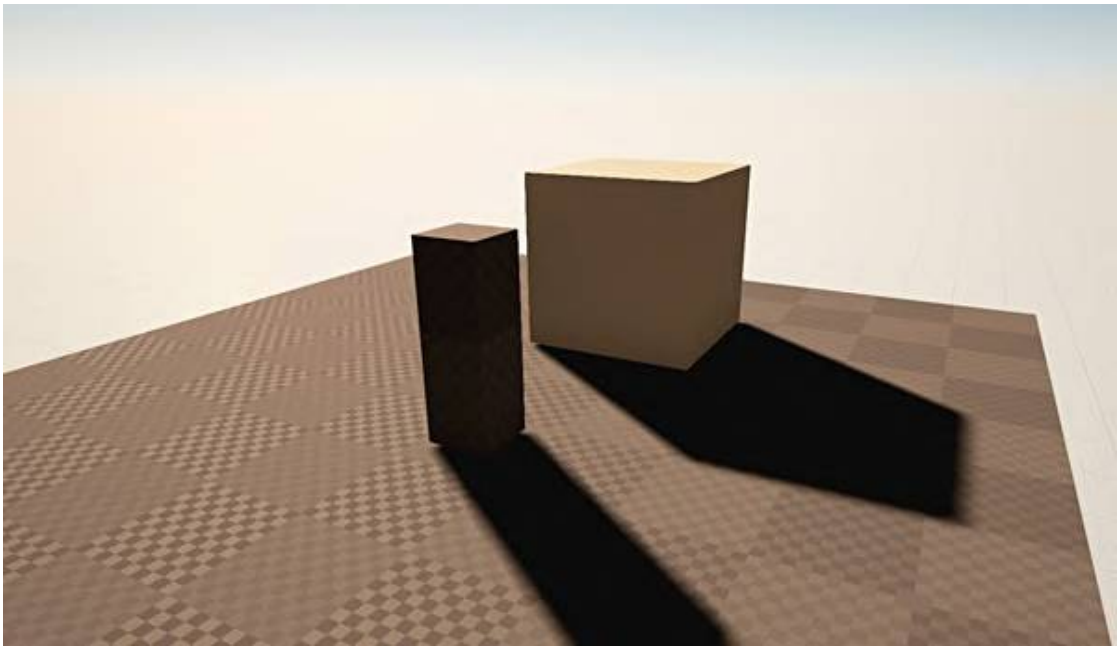
You could continue to insert other static meshes the same way. But let's move onto the next step - inserting a SkyLight.

SKYLIGHT

SkyLight is an additional light actor that illuminates indirectly lit areas; areas that aren't being reached and lit by Directional Light.

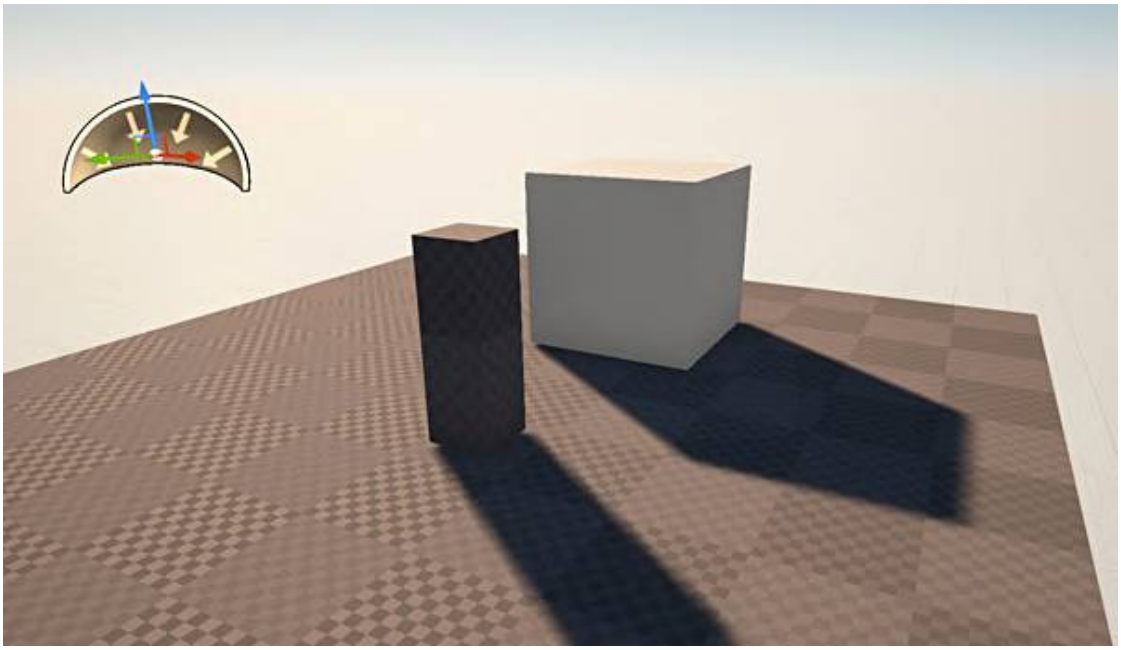
Best way to show you what the Sky Light does is to look at these two images.

This image is the same scene without a SkyLight:



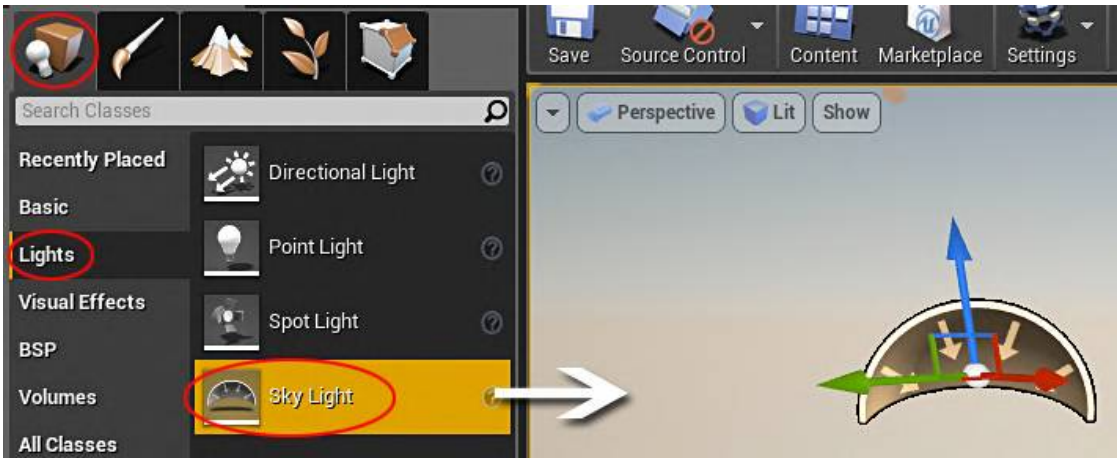
You can see that our shadows are black and with this type of lighting scenario, you should have a lot of bouncing blue light from the sky.

This image now contains a SkyLight:

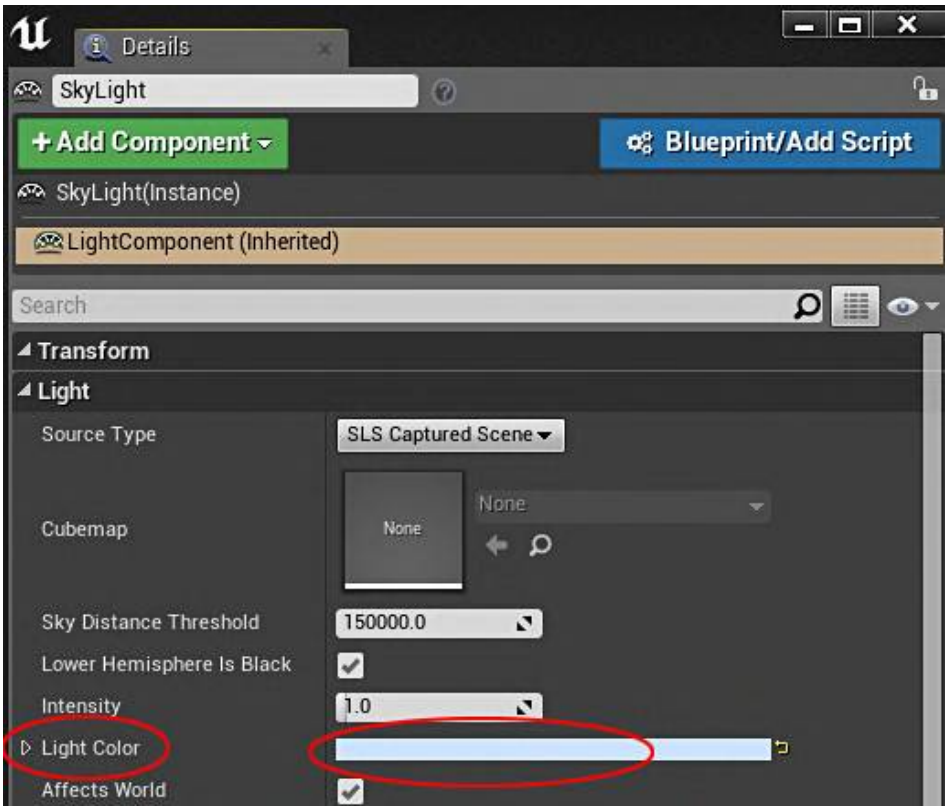


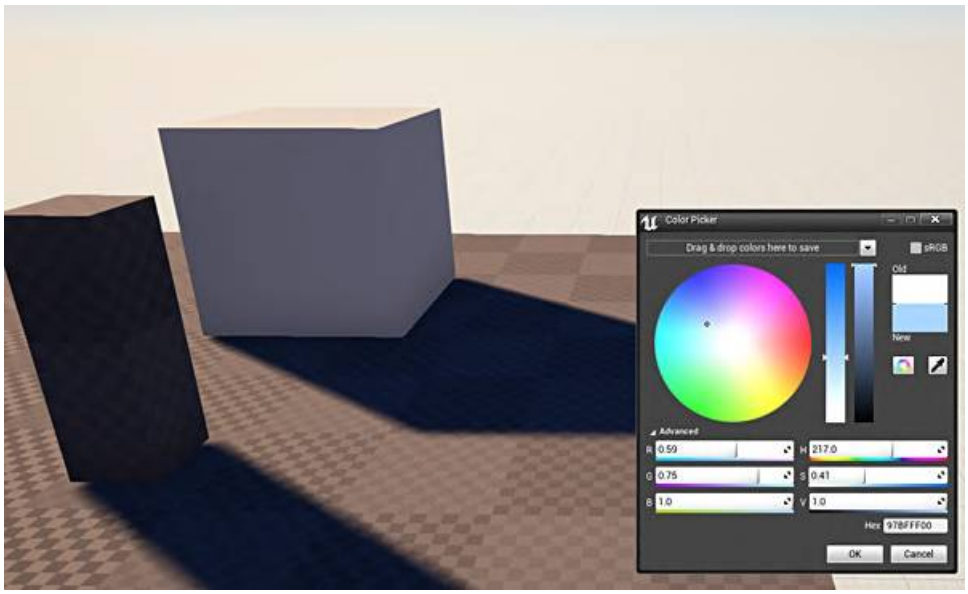
The areas in shadow are no longer black and it looks a more realistic with indirectly lit areas receiving blue hue from indirect lighting, from the SkyLight. For SkyLight to work you need to have either a sky sphere or Atmospheric Fog actor, this is important.

Go to Place Modes and under Lights, left click and drag SkyLight into perspective viewport:



To change the indirect lit areas with different color, select the SkyLight and in Details panel, change Light Color property:

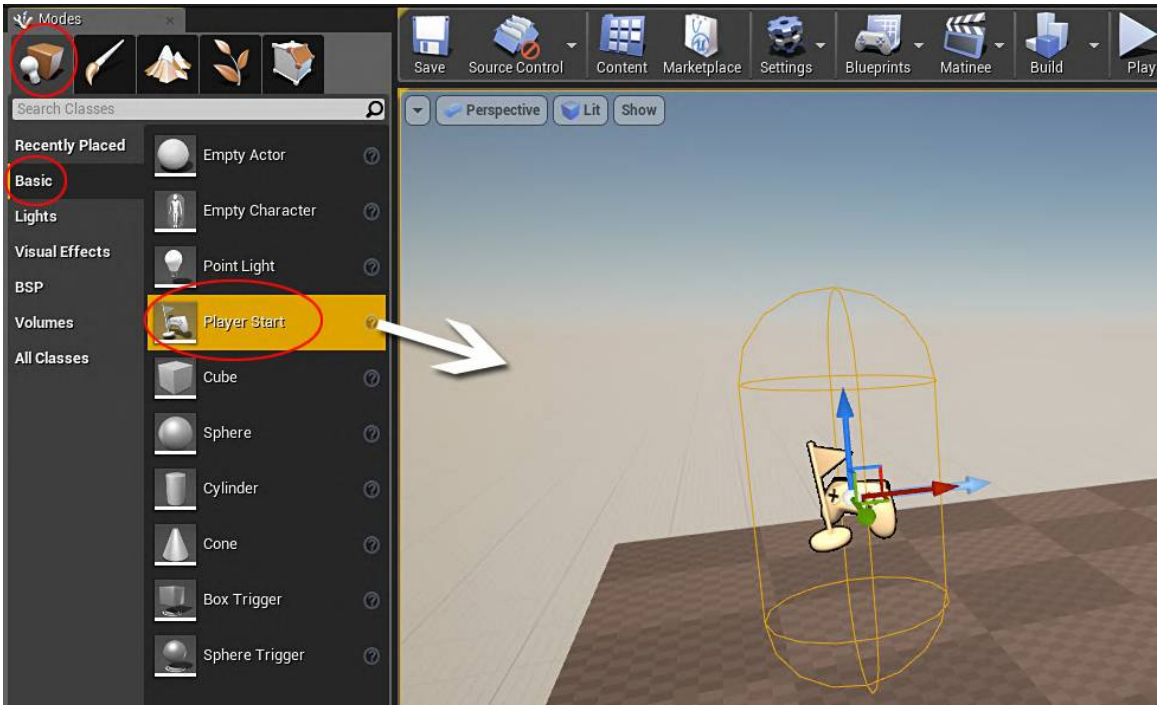




PLAYER START

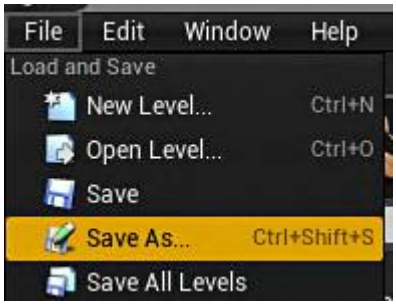
Player Start specifies a spawning location for the player inside the game. Every level should have one player start.

Go to Place Modes and switch to Basics tab and left click and drag Player Start into Perspective Viewport onto the ground plane:

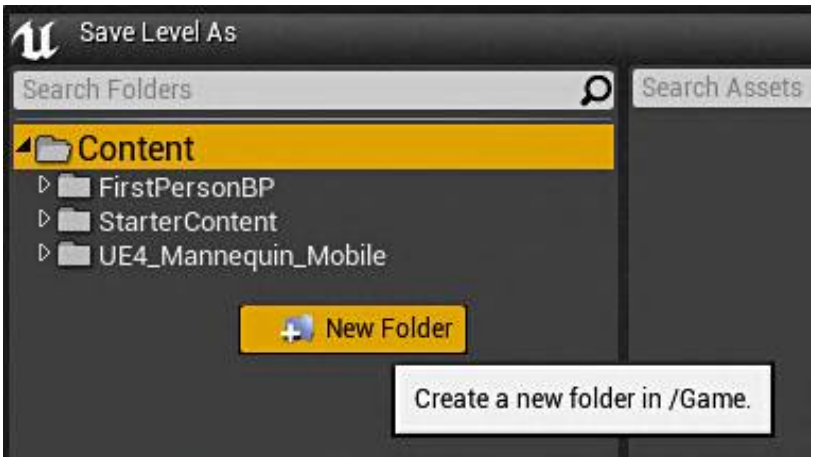


SAVE YOUR LEVEL

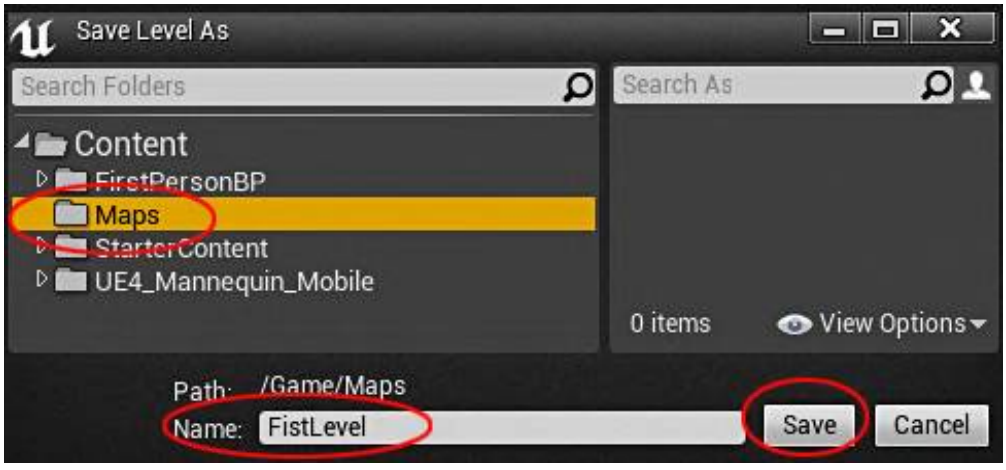
We've been working on our level for a while. Let's save before we lose any of the work. Go to File > Save As:



For better organization, create a new folder to save all the maps into. Right click and choose New Folder and name it "Maps":

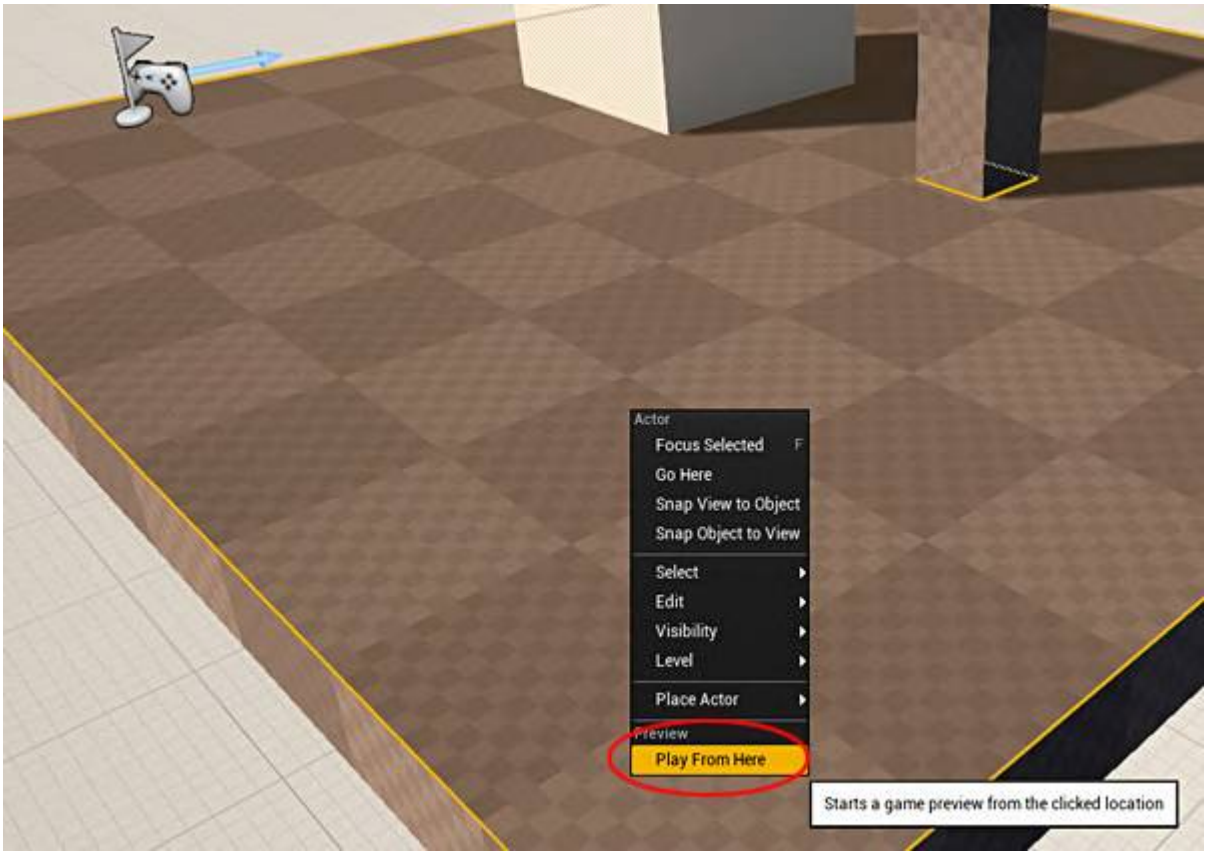


Select the Maps folder, name your level and click Save:

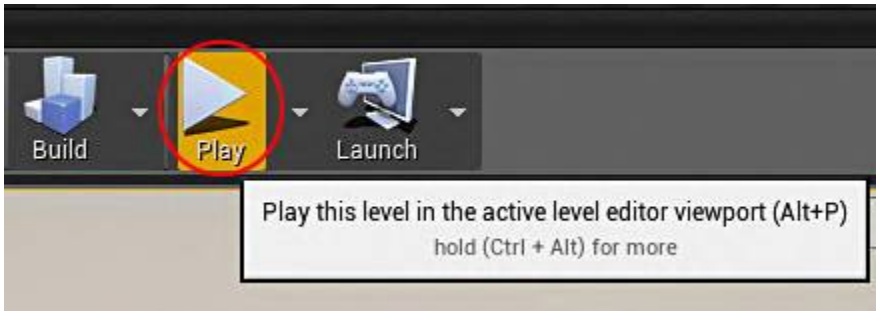


TEST YOUR LEVEL

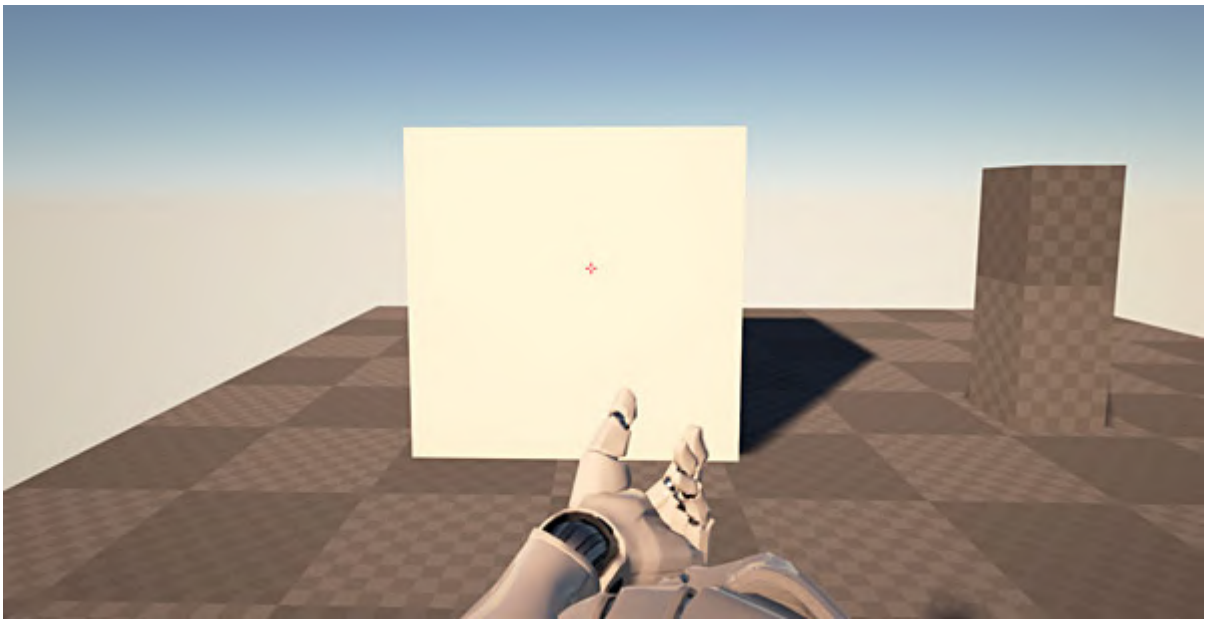
You can play test your level right from the editor. In perspective viewport, right Click and choose "Play From Here". This will spawn you from the location where you right clicked:



If you press on the "Play" icon at the Top Toolbar, it will spawn you from the Player Start actor inserted earlier:



You will spawn inside the level and as a player character depending on which game template you chose at the beginning of project set-up:



Press "Esc" to return to editor mode.

BUILD ALL

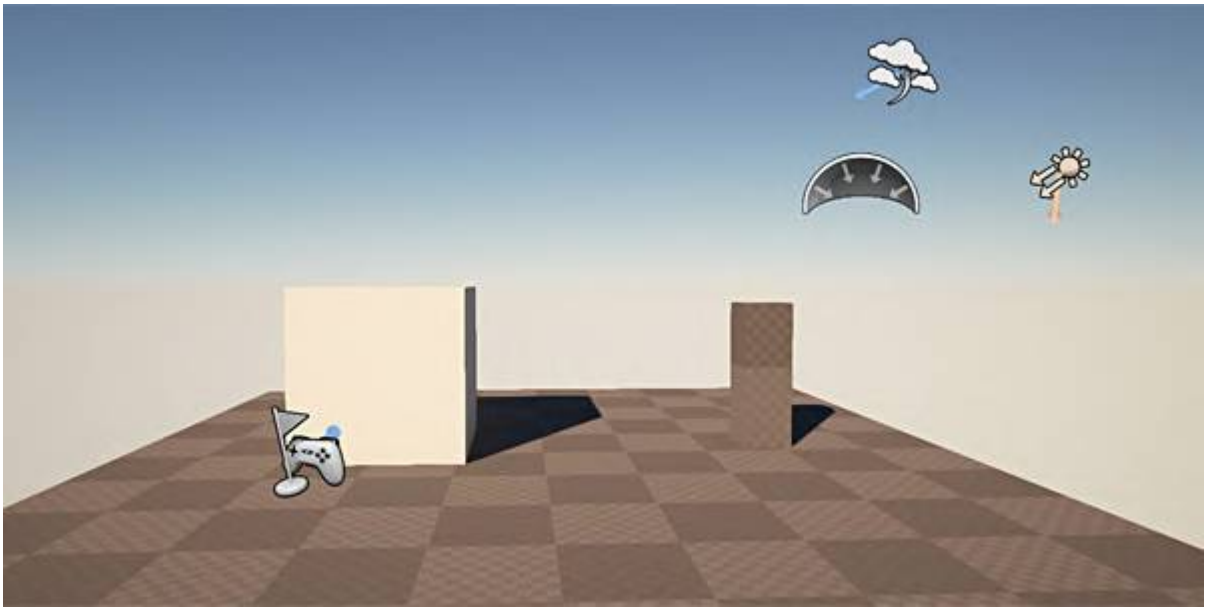
Building a level calculates and renders lighting, geometry and navigation. It shows how the environment really looks. Up till now it only shows us an approximation or preview lighting.

Go up to Main Toolbar and use the drop down menu to set Lighting Quality to Production. We've used Preview Quality but towards the end of your environment creation, we need to have everything rendered at the highest quality possible.

After setting to Production, hit Build:

WHAT'S NEXT?

You have now created a very simple level, complete with a player start, a light, Static Mesh, fog and we build on production and saved the level.

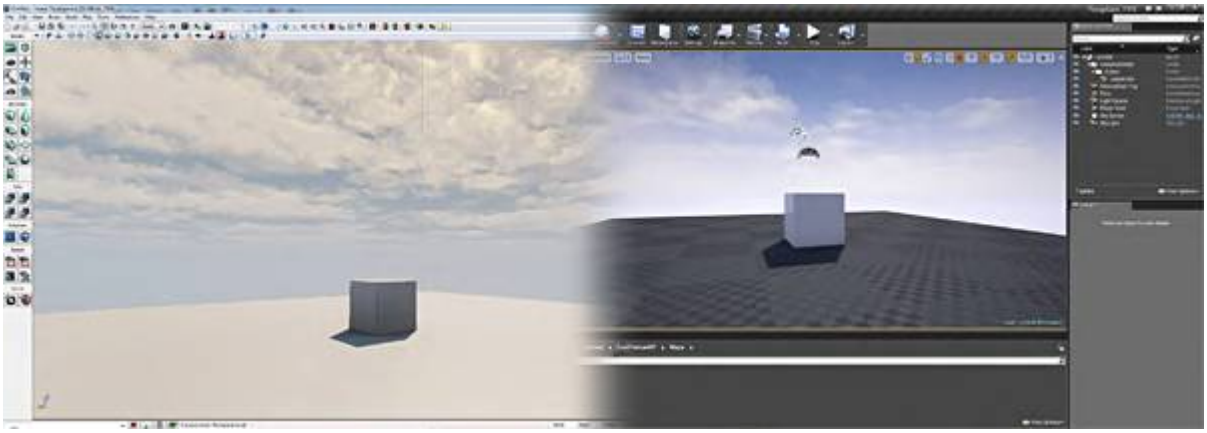


Of course there is a lot more to this than what we just covered, but these are the basic steps of creating a level.

You are now set to explore advanced topics within UE4 and begin to create your own, more complex world.

For more tutorials visit WoLD Unreal Engine 4 Tutorials page.

SECTION 4:
26 STEPS TO TRANSITION FROM UDK TO UE4



How do you switch or transition from UDK to UE4?

If you have been working with UDK (UE3) for a few weeks, months or years, there will come a time when you'll want to switch to Unreal Engine 4. The updated tools, easier workflow and new improved tech are too appealing to pass up. And it is best time as any to make this transition.

Epic has focused on making Unreal Engine 4 the game engine to use. It is now free (no more subscription fee) and constantly being updated. While UDK is being phased out and will no longer receive any updates.

With the experience and knowledge you have from UDK, it will be easier to make the jump to UE4. Most of the functions, principles and workflows haven't changed, they just received an upgrade.

In this tutorial you will learn how to make the quick transition and how to get started with UE4 when you are coming from UDK (UE3).

UE4 is very easy to get into, even for a complete beginner.

Here is a broad overview of just a few improvements and upgrades:

- Overall appearance of the editor has been changed and updated; the system interface is completely fluid and you can customize it to exactly how you want it
- Scripting/Coding is now in C++, no more Unreal Script

- Kismet visual scripting language has been replaced with a much more powerful tool called Blueprints
- Material creation is different than what you are used to in UDK - UE4 now uses PBR (Physically Based Rendering) workflow
- Lighting has been improved
- Bringing assets into UE4 is a simpler process
- Various game templates to get your game started
- Ability to create and manage your projects

These are just a few of many changes.

Before we get started there is one question that often comes up.
"Should I start with UE4 right away or should I learn UDK first and then switch to UE4?"

At this point I would recommend starting with Unreal Engine 4 and not worry about UDK at all. There are some users who still prefer UDK over UE4. Even I still love UDK, as I have spent so much time with it. But as a complete beginner, start with Unreal Engine 4.

Now, let's get started with how quickly transition from UDK to UE4.

1. FREE UNREAL ENGINE 4

Unreal Engine 4 is now free. It started off as a monthly subscription service, but since then Epic has let that pricing go, making it free for everyone. So just like UDK, you can freely download UE4.

Tutorial: UE4 - How to Download and Install Unreal Engine 4

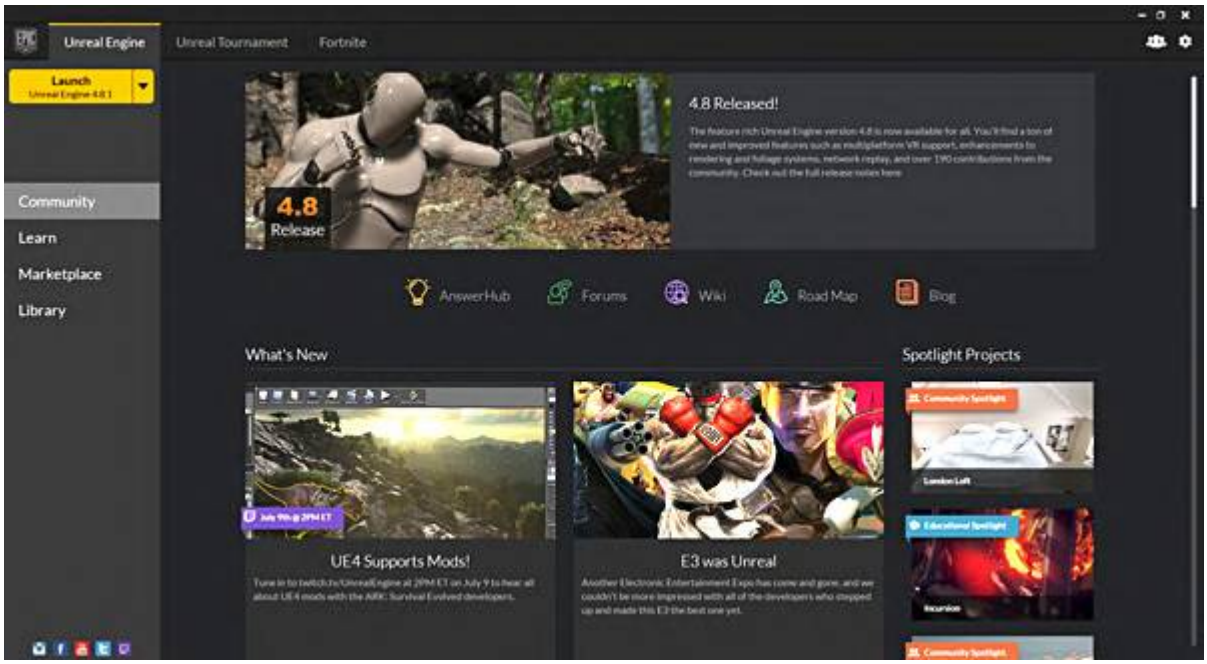
As for royalties and releasing your own game with UE4, visit here for all the details.

2. UNREAL ENGINE/EPIC GAMES LAUNCHER

In UDK, you would download a specific version of the engine as a large exe file and install.

In UE4, the download and install works a bit different.

First, you download a small installation file that will set up Epic Games/Unreal Engine Launcher. This is a portal through which you download and install any version of the engine, old and new. Any future updates will be applied and downloaded through this Launcher.



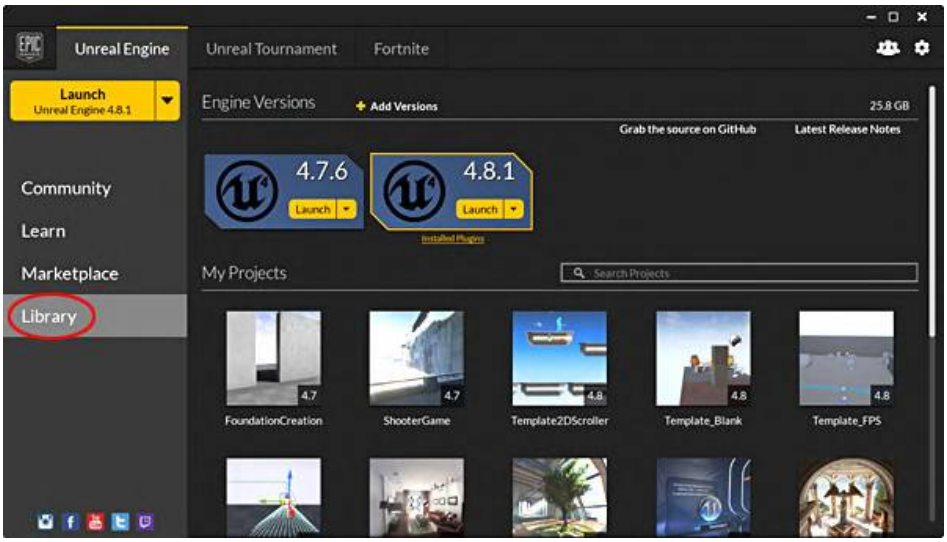
So you no longer have to check if there is a new Unreal Engine 4 version out, the Launcher will keep everything up to date and notify you what's new. Through the Launcher you will also have the ability to create and manage your projects, purchase Marketplace content for your projects, download examples and game engine content from the Learn section.

3. PROJECT MANAGEMENT

In UDK, to create a project you would launch the editor and start working on your project. There was no way to keep various projects separate other than setting up folders ahead of time and saving all the maps, packages and content into it.

In UE4, you have to launch a project you will be working on in order to open the editor. To work on another project, you have to launch another instance of the editor for that project.

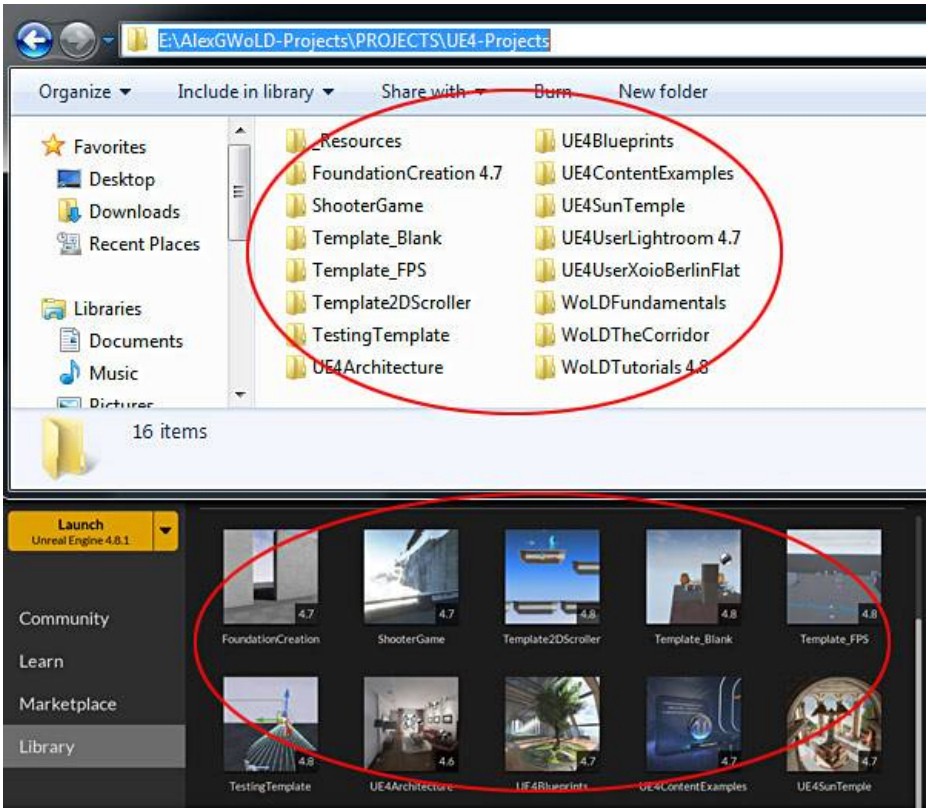
All of your current, worked on project will be under the Library tab:



Before starting UE4 editor you would define a project, and then launch the UE4 editor with all the files associated with it for that project.

Tutorial: [UE4 - How to Create, Start and Open Your First New Project]

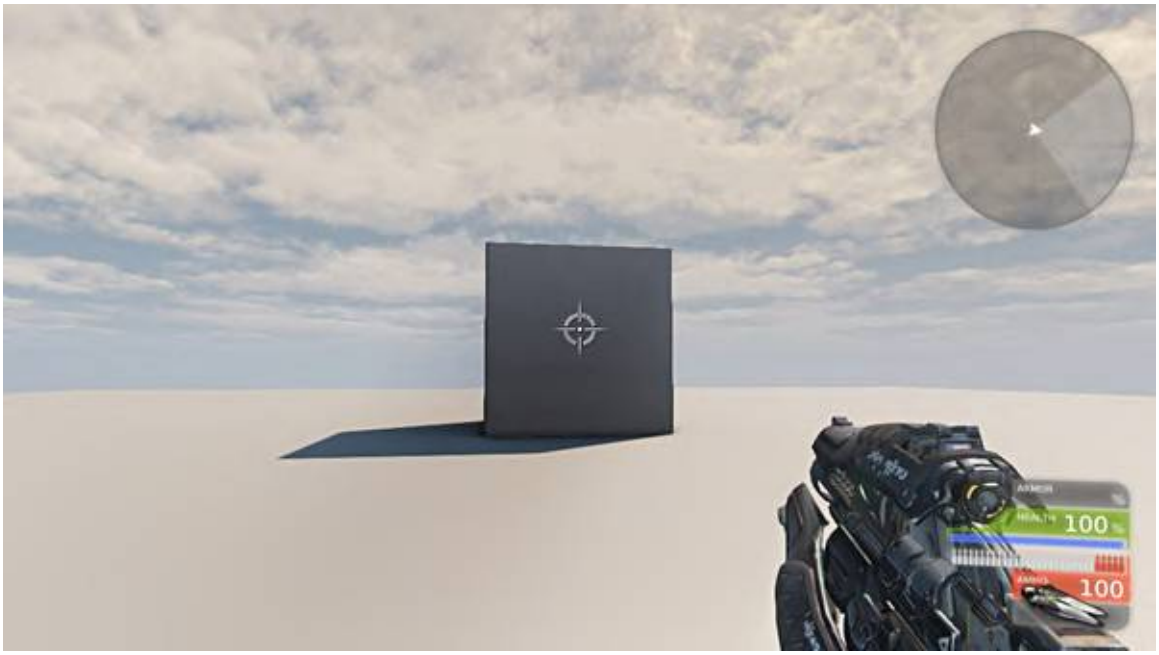
Each Unreal Engine project will exist in its own directory and will contain its own project files.



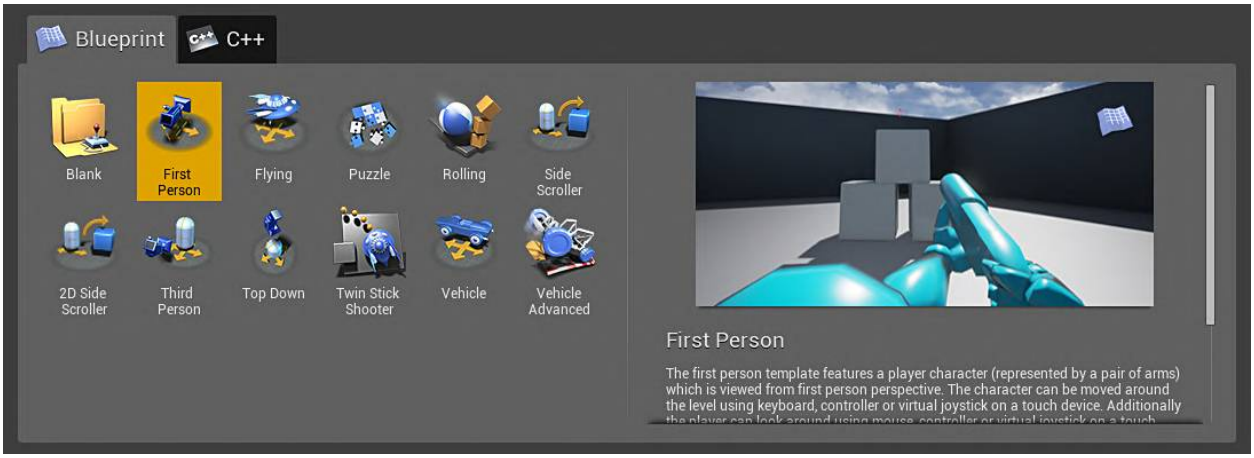
Unreal Engine/Epic games Launcher is also where you manage and open any project that you are working on.

4. GAME TEMPLATES

In UDK, you would start with first person game mode. If you wanted third-person, top down or side-scroller game, you had to set this up yourself. There was no choice for game templates that gives you basic functionality of a game you want to create.



In UE4, you get to choose to start any project with a game template. You can choose to have C++ or Blueprint template.


Blueprint
C++
Blank
First Person
Flying
Puzzle
Rolling
Side Scroller
2D Side Scroller
Third Person
Top Down
Twin Stick Shooter
Vehicle
Vehicle Advanced
First Person
The first person template features a player character (represented by a pair of arms) which is viewed from first person perspective. The character can be moved around the level using keyboard, controller or virtual joystick on a touch device. Additionally

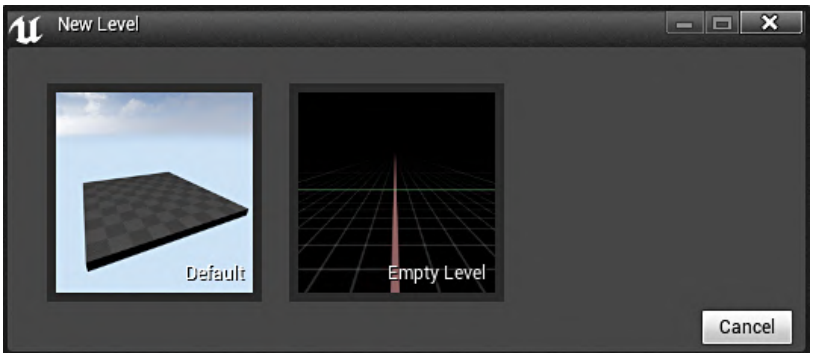
Then, you choose first-person shooter, third-person, side-scroller or any other available game template which gives you a starting point for basic functionality to build and expand upon for your own game.

5. NEW LEVEL TEMPLATES

Level template gives you few necessary actors to begin creating the game world without having to insert these actors yourself. It is a very quick and efficient way to get a level started.

In UDK, if you go to File > New Level, a menu pops up to choose between various map templates. You had 4 templates for different time of day and one blank map template:

In UE4, you still have new level templates, but as of right now there are only 2 templates to choose from. Default and Blank Map:



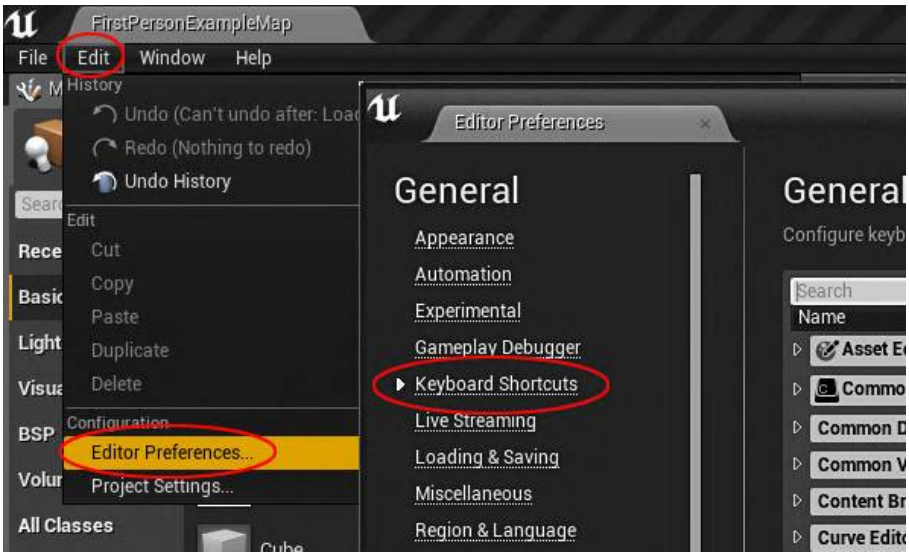
The current single template in UE4 is enough to get started. Perhaps later we'll have more additional time of day templates.

6. KEYBOARD SHORTCUTS

Keyboard shortcuts are almost identical between many common functions from UDK to UE4, but not all.

For example **in UDK** you have Alt+W (Static Mesh Visibility) or Alt+X (Maximizing/Restore Active Viewport). In UE4, these don't exist.

But **in UE4**, it is extremely simple to add/update shortcuts by going to Edit > Editor Preferences > Keyboard Shortcuts:



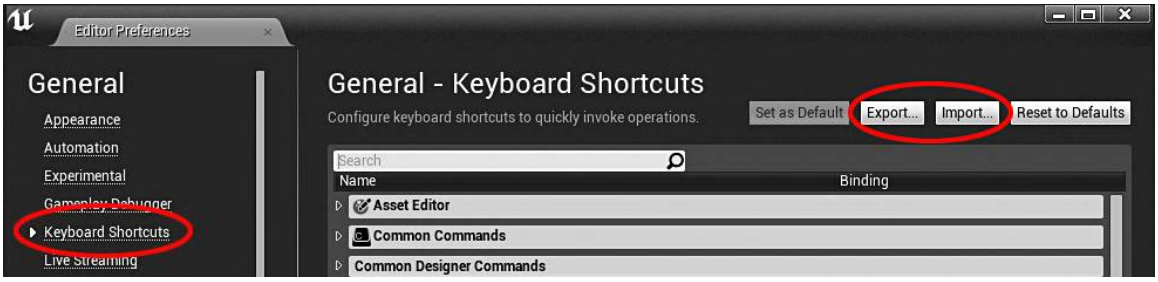
Search for a shortcut and enter the keys for that shortcut to use:

Important note about shortcuts in UE4 is they do not transfer from one project to another.

All the Project/Editor Settings are self-contained within a project. So if you created a new project, you would have to re-assign all the shortcuts keys again.

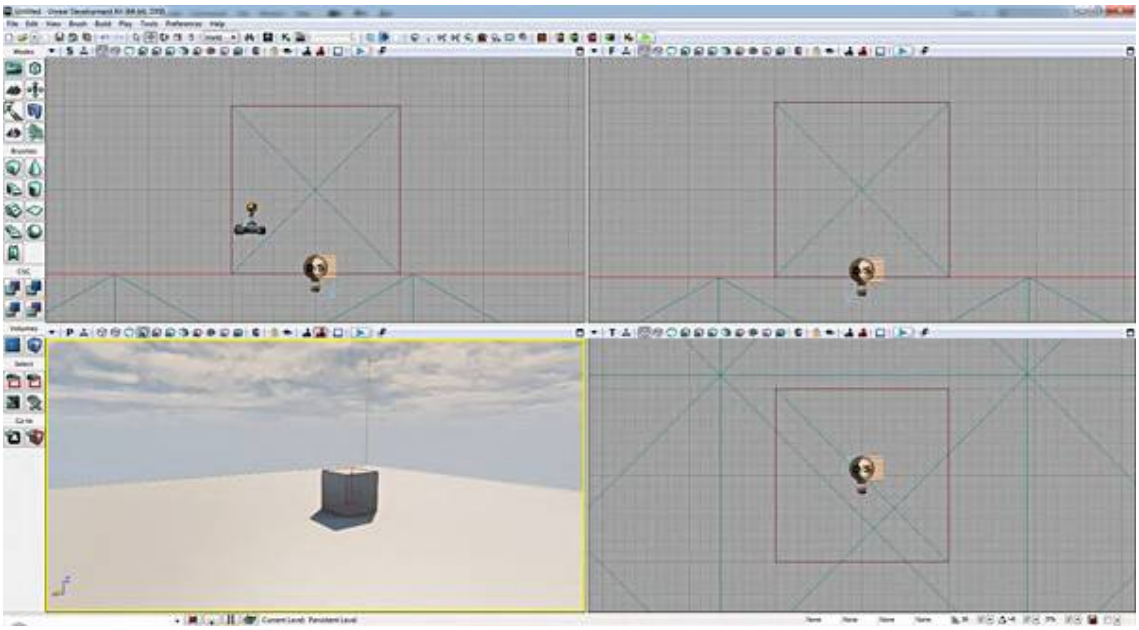
Solution is to export all the shortcuts to a file and every time you start a project, import this file to get access to your shortcuts, every time.

To export or import keyboard shortcuts, go to Edit > Editor Preferences > Keyboard Shortcuts and press Import/Export:

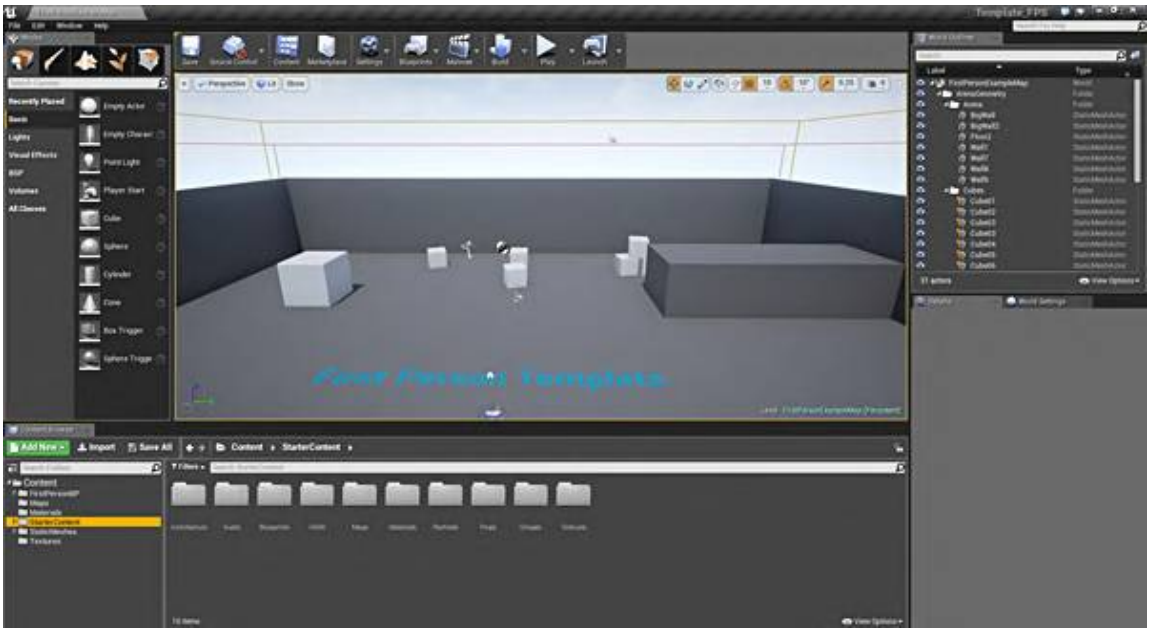


7. TABS AND FLUID INTERFACE

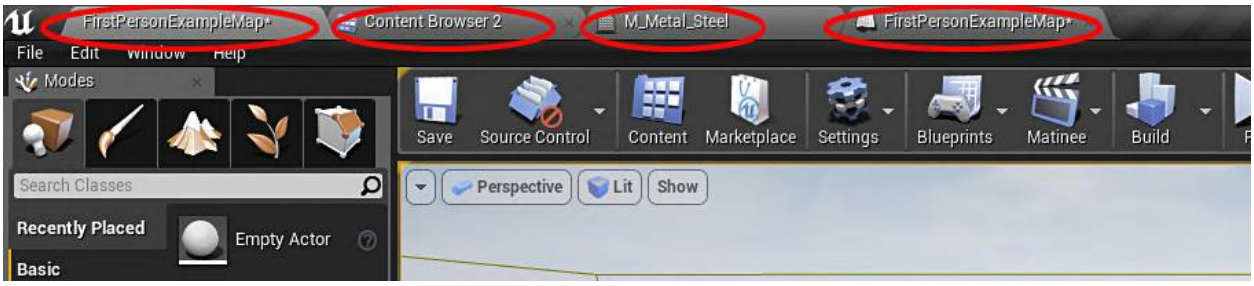
UDK interface contained a lot of buttons, icons and menus. As a complete beginner, it was overwhelming to make sense of it all.



The entire UE4 interface has received a complete overhaul - from colors, to panels to functionality. Everything is now fluid. You can change and customize the entire interface to exactly how you like it. Resize or drag and drop any panel within the editor to modify the interface.



Another useful function I like is the ability to have tabs - just like a web browser. You can have multiple tabs running across the editor. Dock these tabs anywhere you want so you can have Blueprint, Matinee and Material Editor all in their own tabs to avoid having multiple floatable windows open. You can still have menus as float windows, but tabbing adds additional organizational option.

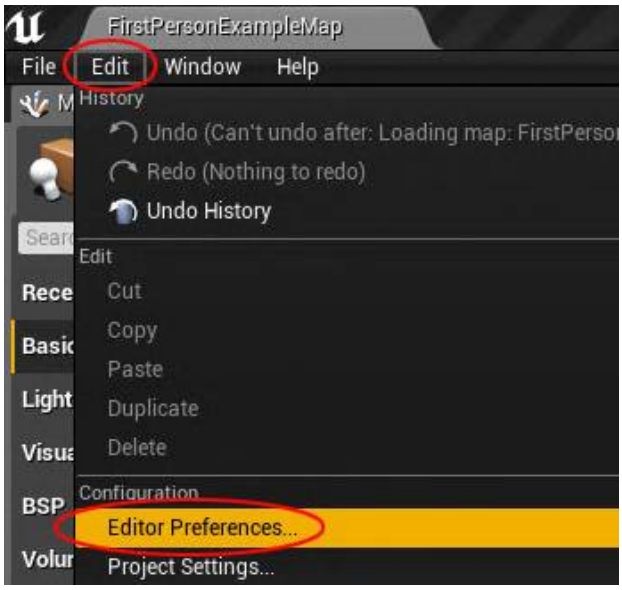


8. VIEWPORT NAVIGATION IN UE4

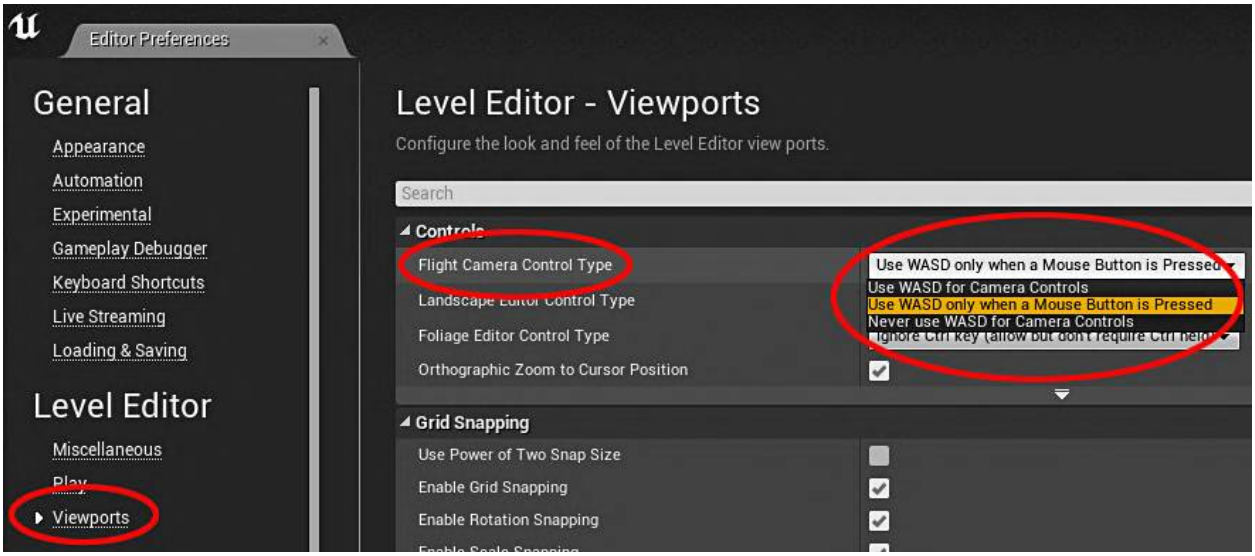
Viewport navigation from UDK to UE4 is almost identical with a few slight changes.

Perspective Viewport Navigation: In UDK, you use WASD keys to move viewport camera in front/back/side directions.

In UE4 you have to press and hold right mouse button while using WASD. You can disable having to press/hold right mouse button for viewport navigation by going to Edit > Editor Preferences:



Under Viewport and Controls section, change Flight Camera Control Type to "Use WASD for Camera Controls":



The rest of the viewport navigation is the same from UDK. Such as holding the right or the left mouse button to look around and move forward and back; and if you hold both left+right mouse buttons you move up/down/side-to-side.

In UDK, if you liked Maya style navigation you had to press L + left or right mouse button in to use it.

In UE4, for Maya style navigation:

- Press and Hold Alt+Left or Right Mouse Button

Also in UE4, press F to center view on selected object and then press and hold Alt while holding and moving the left mouse button. This will rotate around the selected object. Very useful for looking at a single selected object inside the scene.

- F = Center View on Selected Object

Orthographic Viewport Navigation: In UDK, if you hold and move either the right or the left mouse button both enabled panning inside the viewport.

Now, in UE4:

- Left-Click and Drag to Marquee Select Within Orthographic Viewport
- Right-Click and Drag to Pan Within Orthographic Viewport

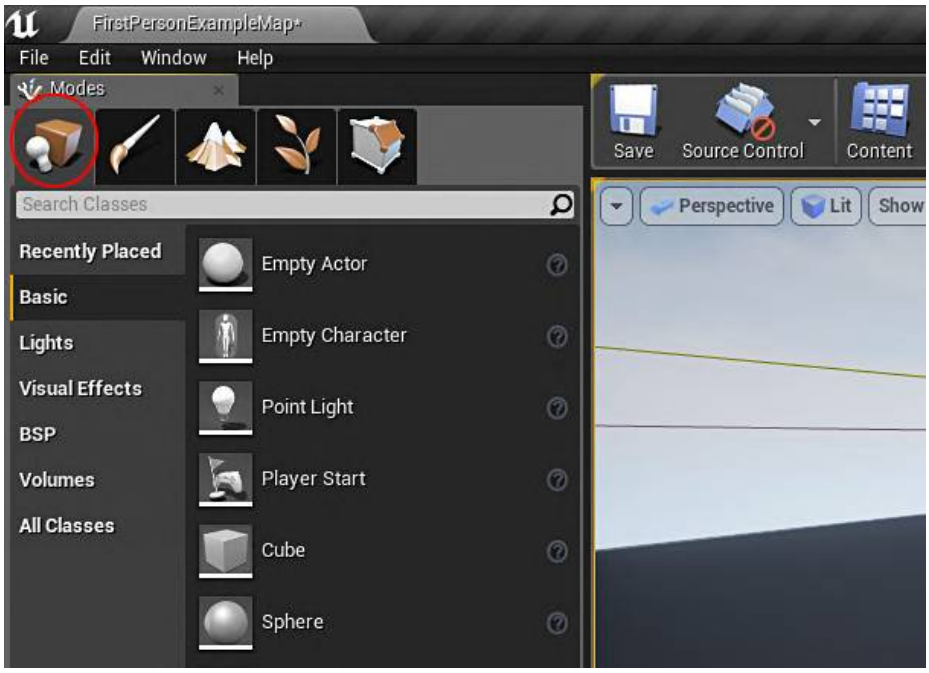
Hold right+left mouse buttons and move will zoom in and out. Same as UDK.

- Hold Right+Left Mouse Button and Drag to Zoom In and Out

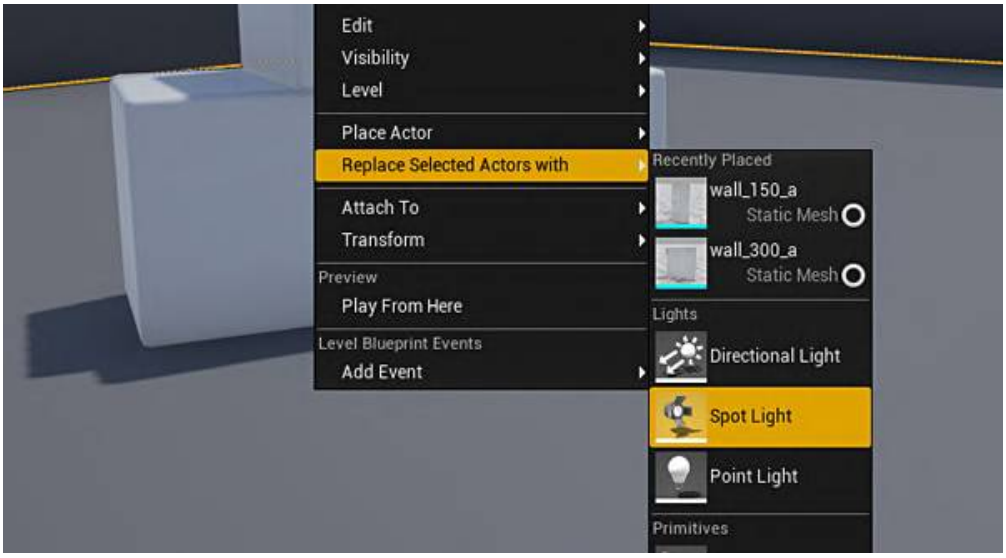
9. MODES PANEL

In UE4, placing objects is a lot easier with the addition of the modes panel (Shift+1).

Within the Basic, Lights, Visual Effects, BSP, Volumes and All Classes tabs you will find most of the needed actors. Left click and drag right into the viewport as in UDK.



You can still right click inside the perspective viewport to insert actors:

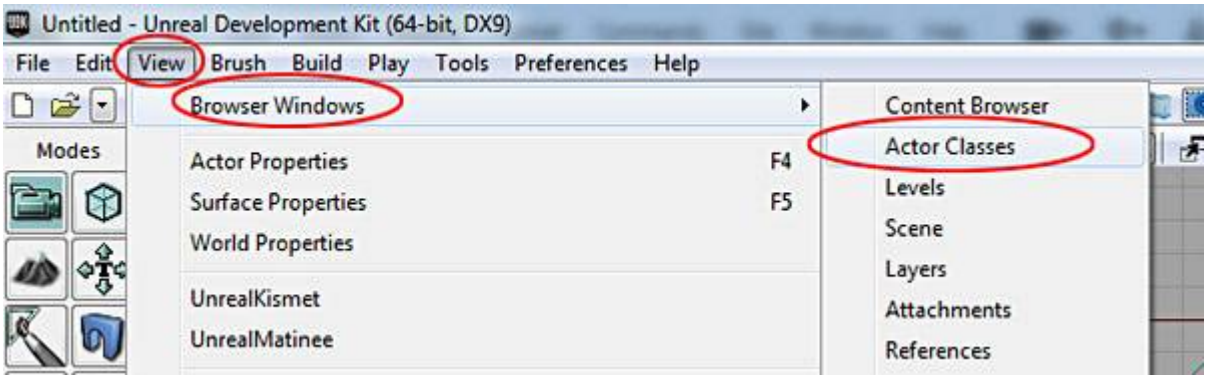


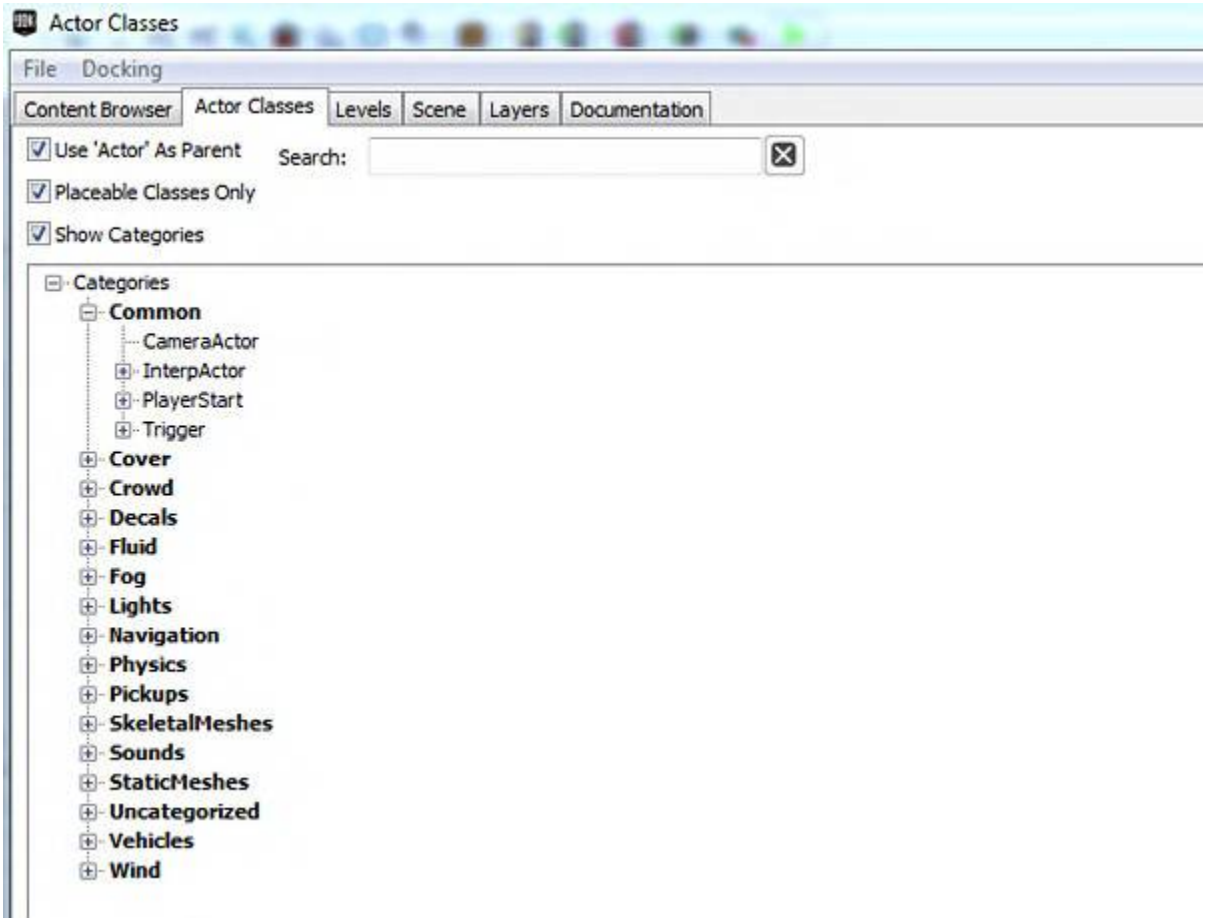
For all the available actors inside UE4 you now have to use Class Viewer.

10. CLASS VIEWER

In UDK, to view all actors you had to access them through Actor Classes menu.

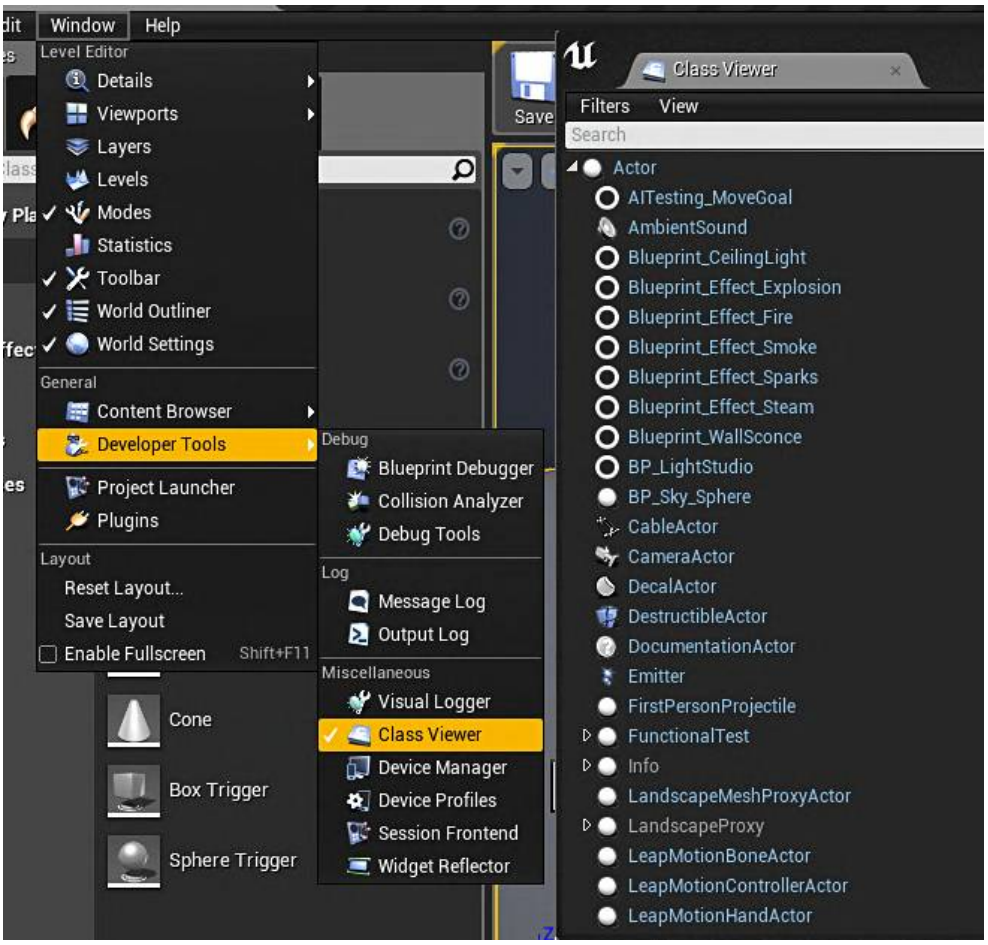
View > Browser Window > Actor Classes:





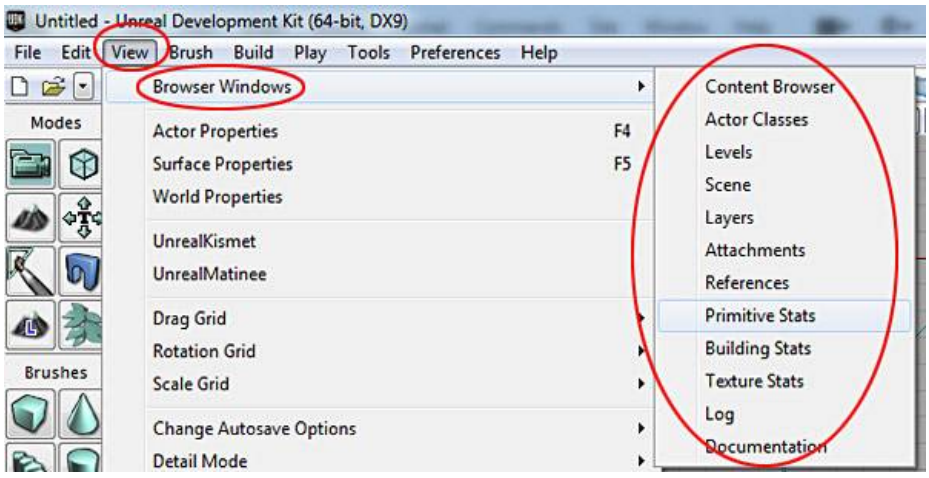
In UE4, Actor Classes is now called Class Viewer.

Go to Window > Developer Tools > Class Viewer:

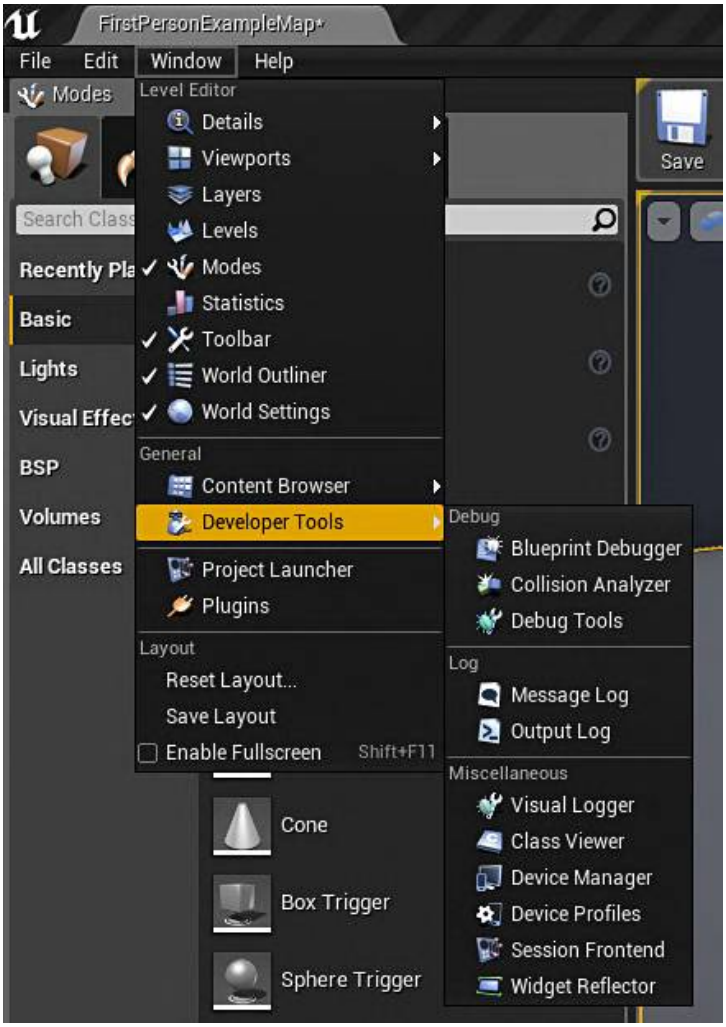


11. DEVELOPER TOOLS

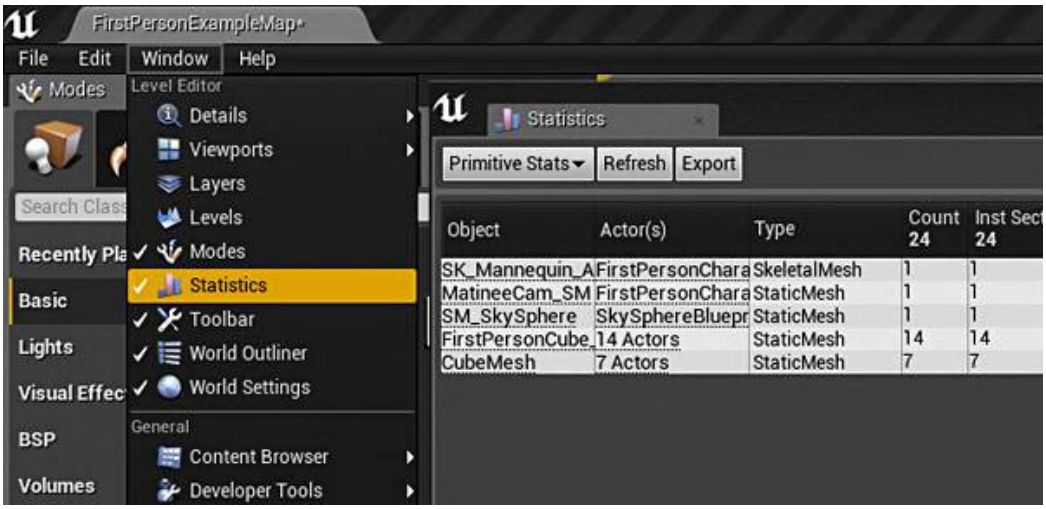
In UDK, you had View > Browser Window, which gave you access to various developer tools such as Log, Texture Stats and Primitive Stats:



In UE4, these developer tools are under Window > Developer Tools:



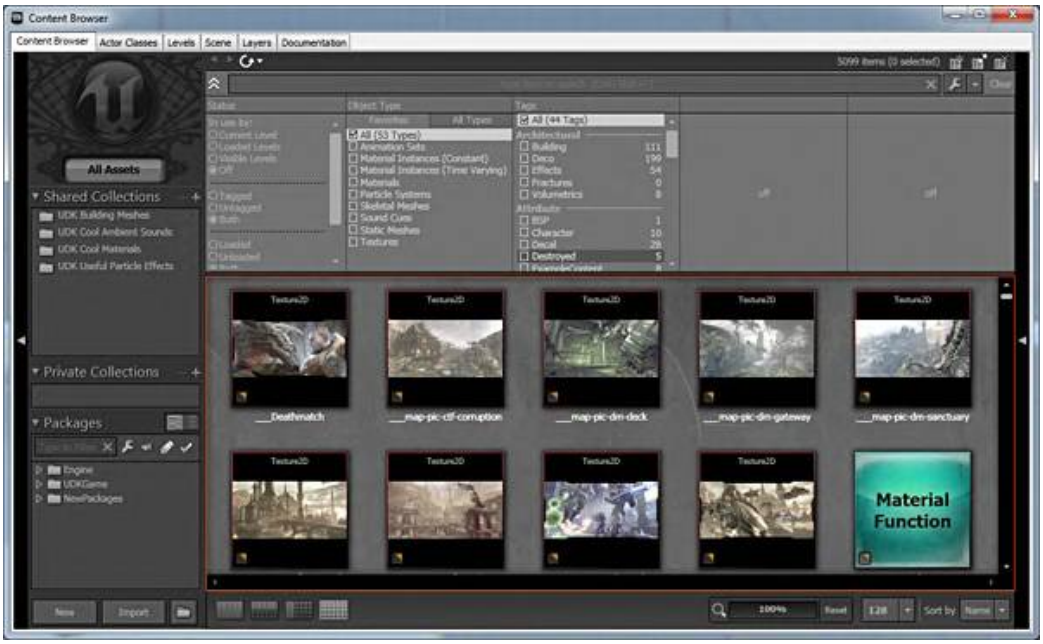
To access Statistics for primitive objects, go to Window > Statistics:



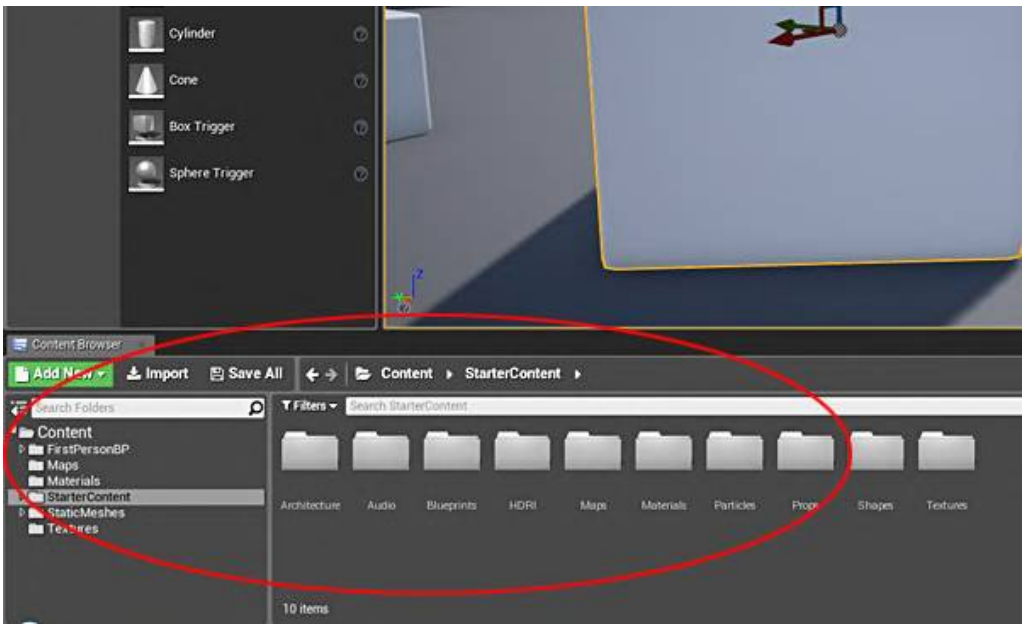
12. CONTENT BROWSER

Content Browser is the content management system in both UDK and UE4.

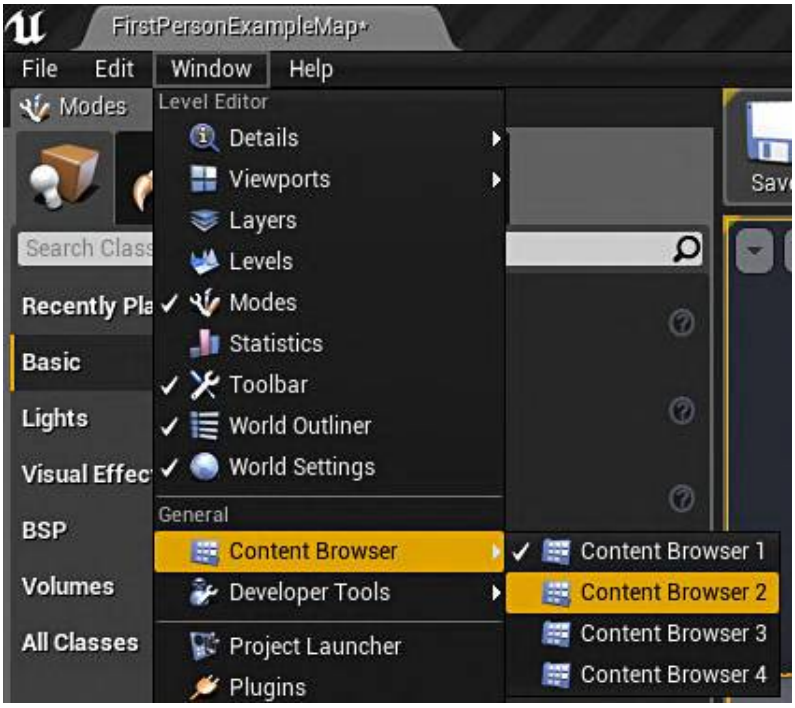
In UDK, Content Browser was a floating window and could be accessed via top toolbar icon or Ctrl+Shift+F shortcut:



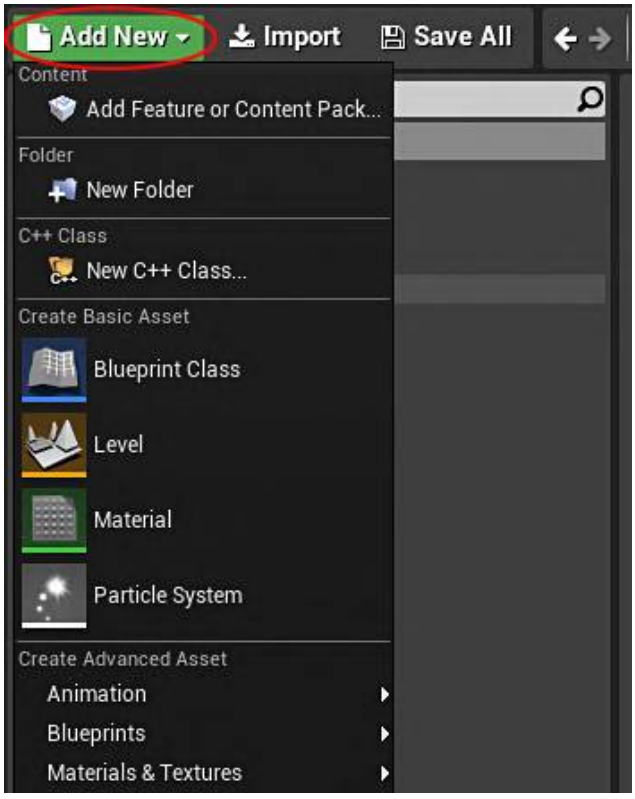
In UE4 Content Browser is now part of the interface, by default it's on the bottom panel of the editor (or on the left for engine v4.7 or older):

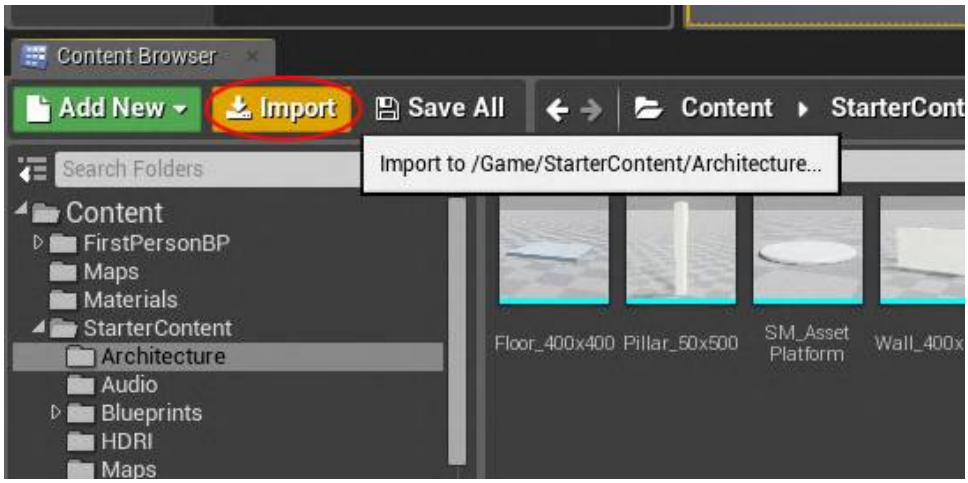


You can open up additional Content Browser windows:

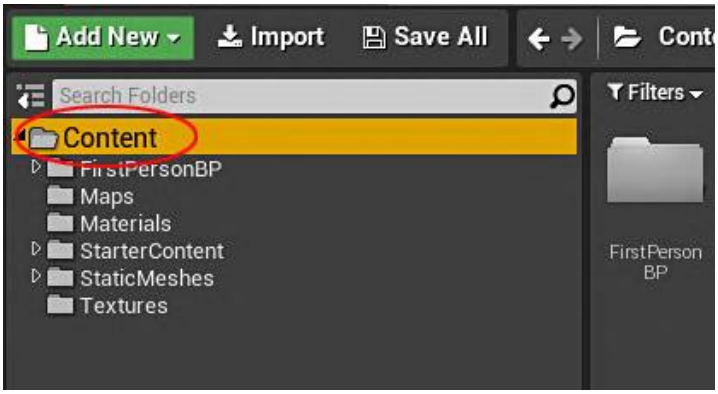


Using the Content Browser in UE4 is more intuitive and easy. You can quickly import or create any object by clicking on Add New:

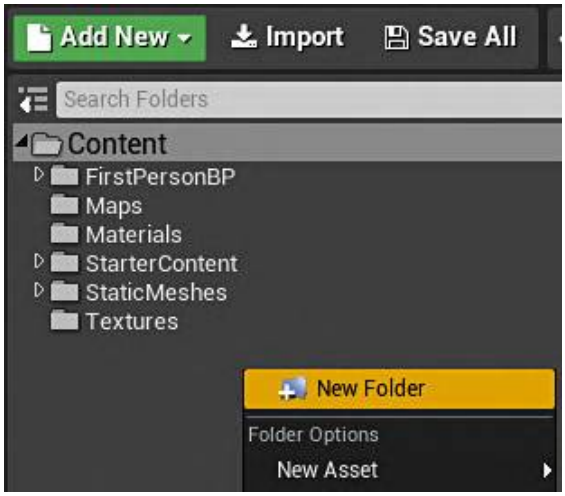




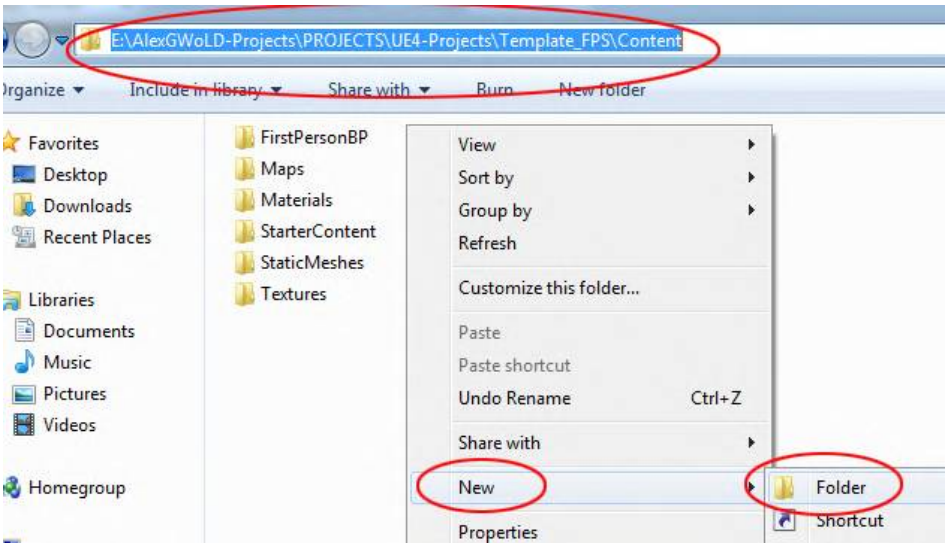
You can drag and drop assets from any folder right into Content Browser. Everything is organized by folders and all of your project assets will be found inside Project Folder > Content:



You can create folders to organize your content inside the Content Browser itself. Right click and choose New Folder:



You can also create folder by navigating to a Project Folder directory, within Content folder. This folder will show up inside Content Browser:

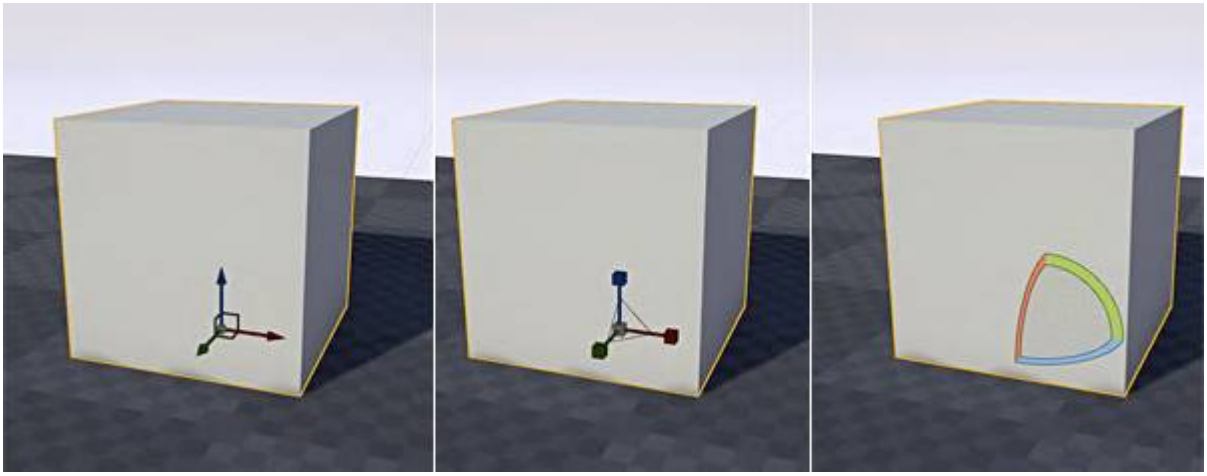


13. WORKING WITH OBJECTS

Working with Actors (objects inside the editor) is similar to UDK.

Placing, selecting, deleting, resizing, moving and rotating are the same.

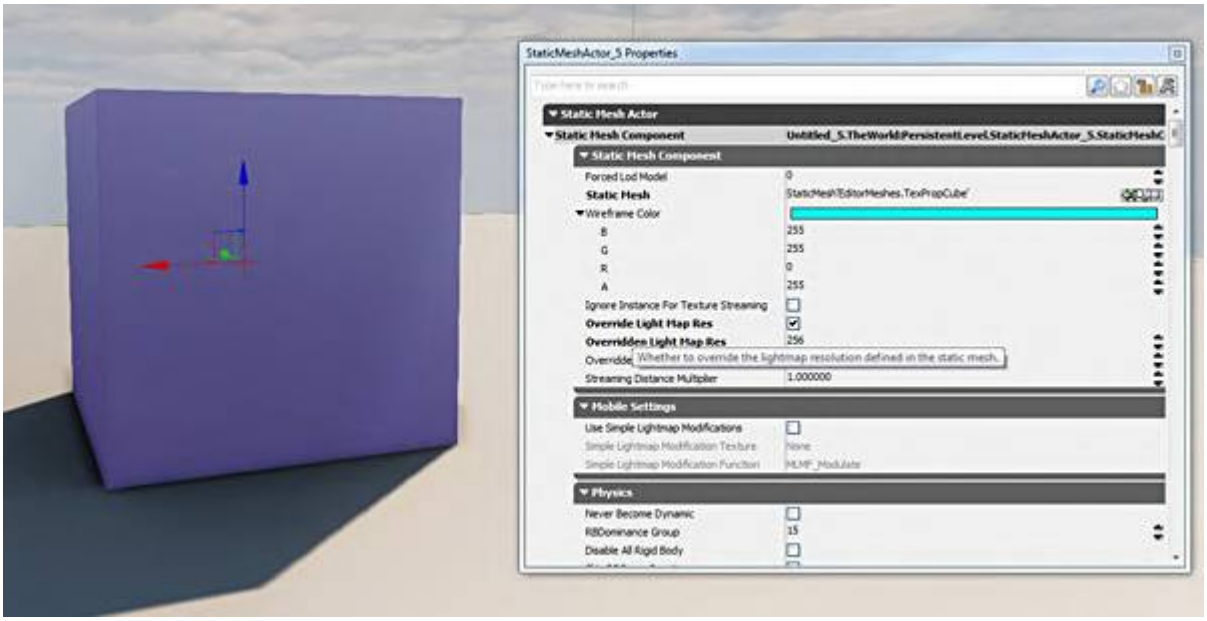
Such as drag and drop right from the Content Browser and using the Spacebar to cycle between move/rotate/scale gizmo.



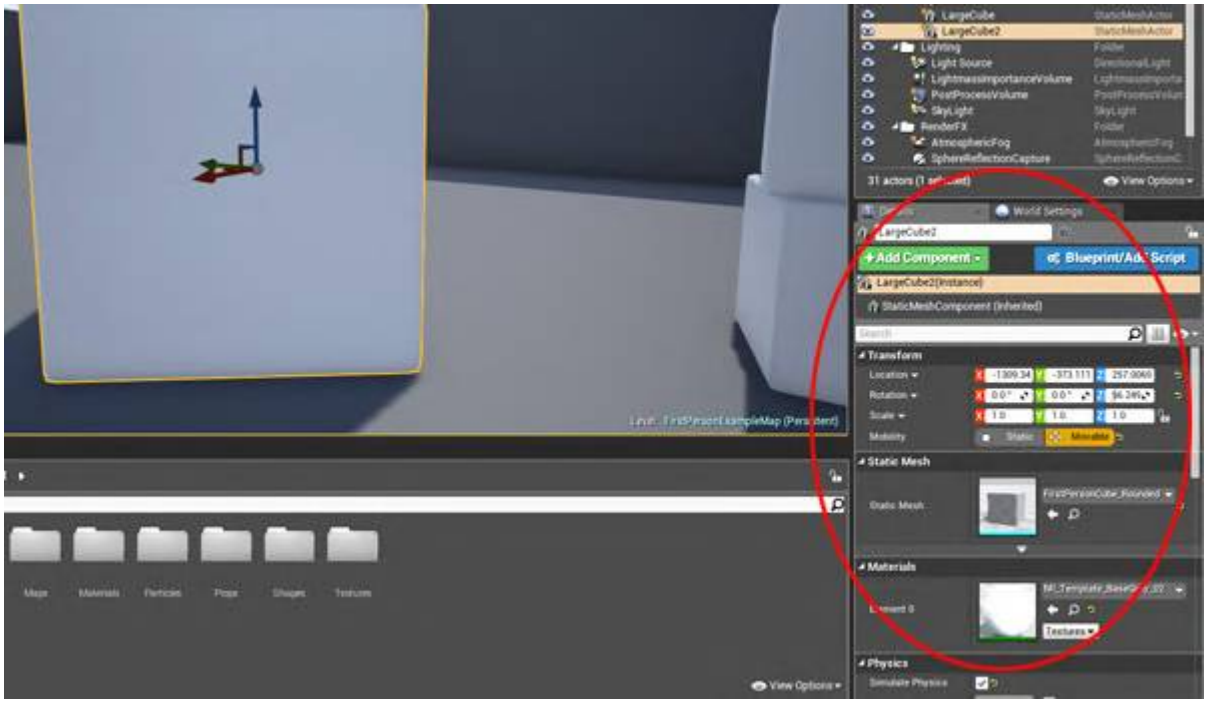
World Outliner panel window gives you additional control over your entire scene and selecting objects inside your level but more on this later.

14. DETAILS PANEL

In UDK, you have to double click to access the object properties. This would always be a floating window.

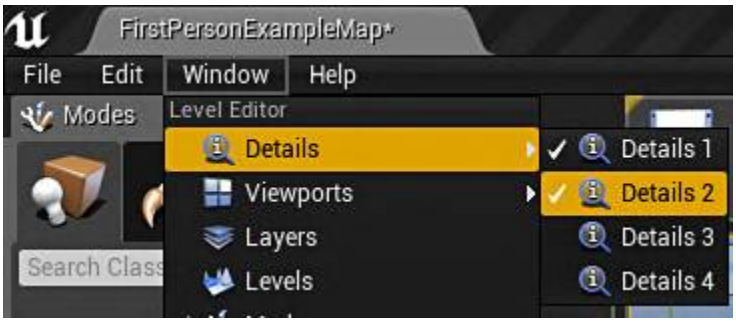


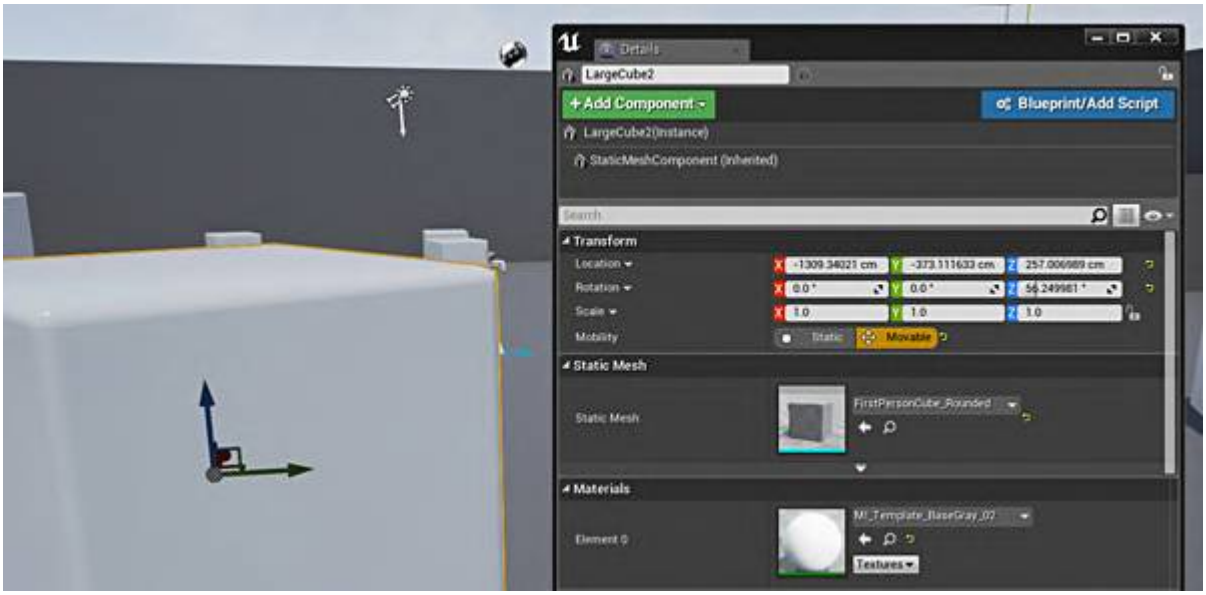
In UE4, object properties are now called Details. This menu is part of the editor and you'll find it on the lower right hand side:



Every time you select an actor/object in UE4, the detail panel will change and show the properties for that selected object.

Of course you could also have Details menu as floating window:

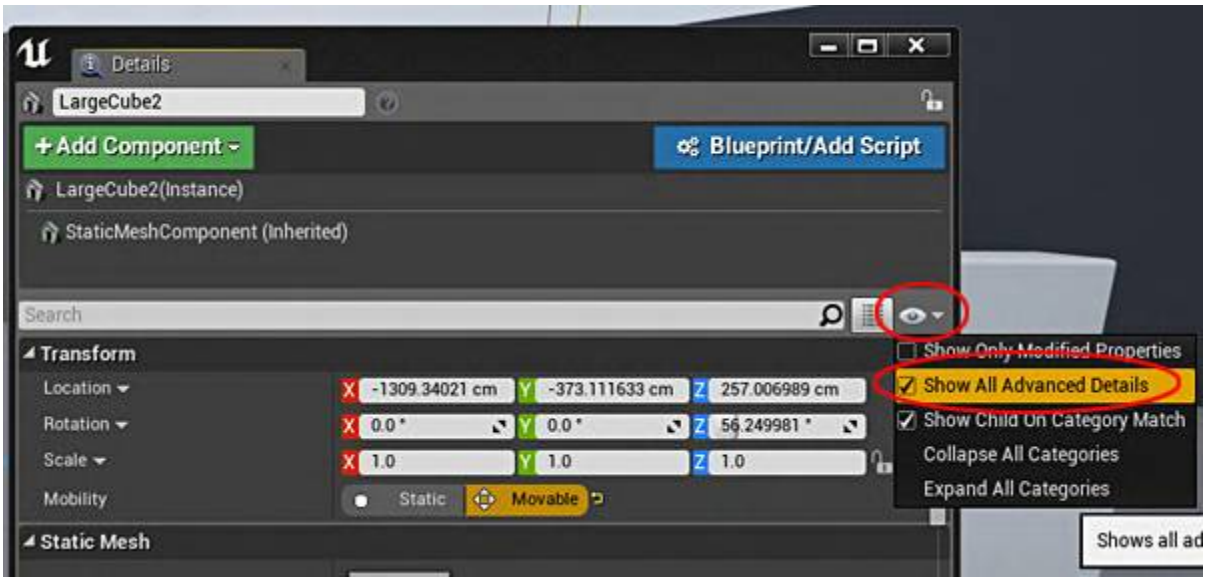




15. SHOW ALL ADVANCED DETAIL PROPERTY

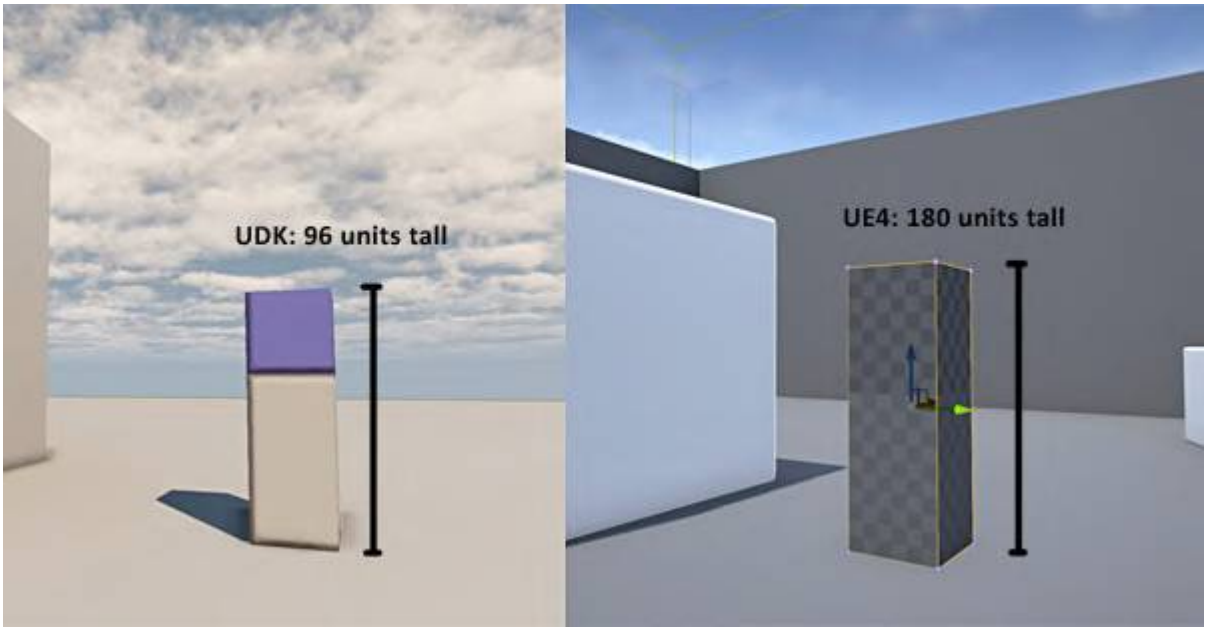
While looking at Object Properties via Details Panel in UE4, you will not see all properties available for that object.

You have to enable “All Advanced Properties” by left clicking on the eye icon to "Show All Advanced Details":



16. SCALE

Scale is different in UE4 from UDK.



- UDK: 1uu = 2cm
- UE4: 1uu = 1cm

Make sure to change working units inside your modeling application to centimeters.

Tutorial: UE4/Maya: How to Set Up Grid in MayaLT/Maya to Match Unreal Engine 4

UE4 Character Scale Dimensions are:

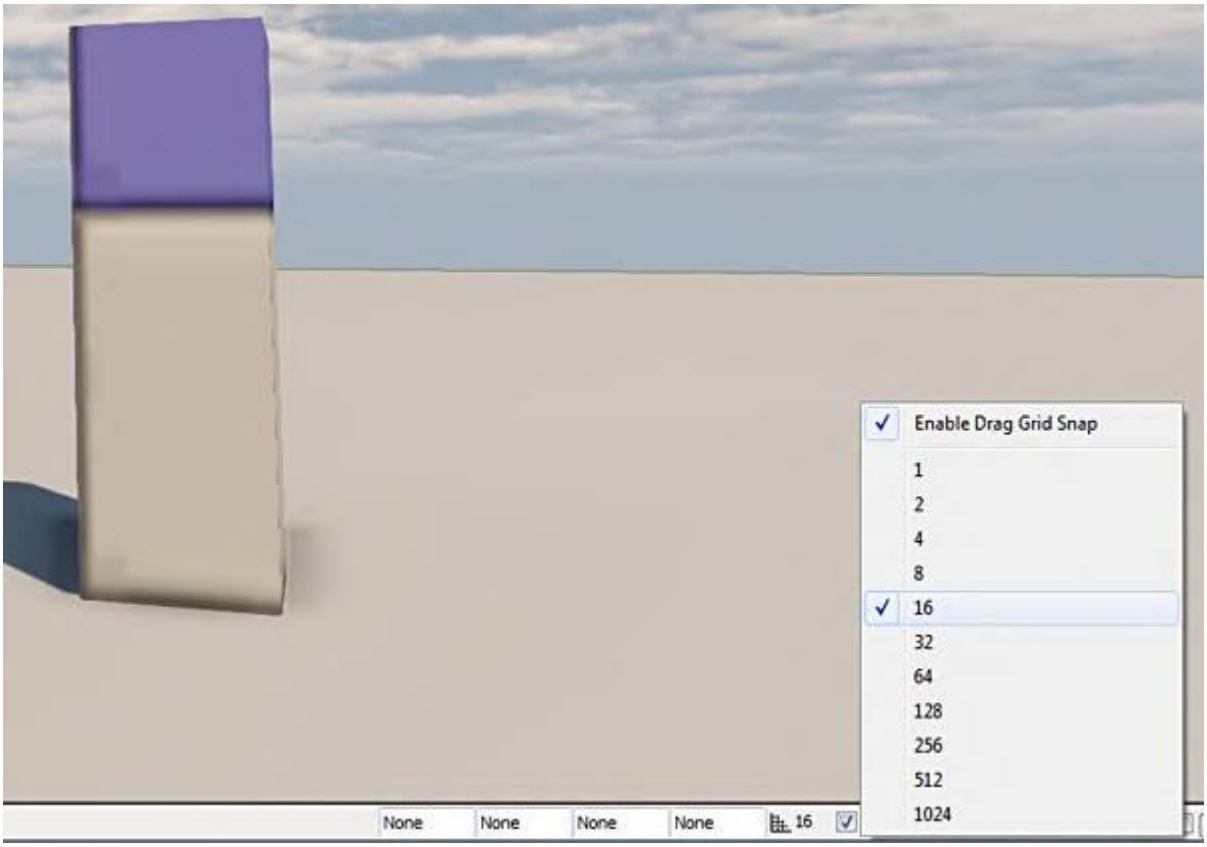
- Height: 180cm/uu
- Width: 60cm/uu
- Length/Depth: 60cm/uu

Tutorial: UE4 - Guide to Player Scale and World/Architecture Dimensions

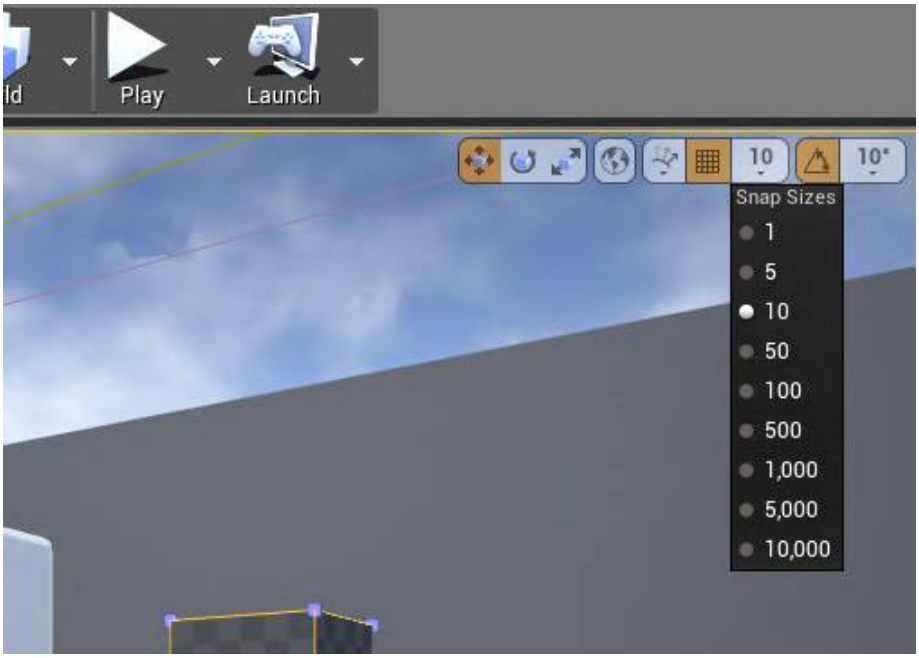
---

17. GRID SETTING SIZE

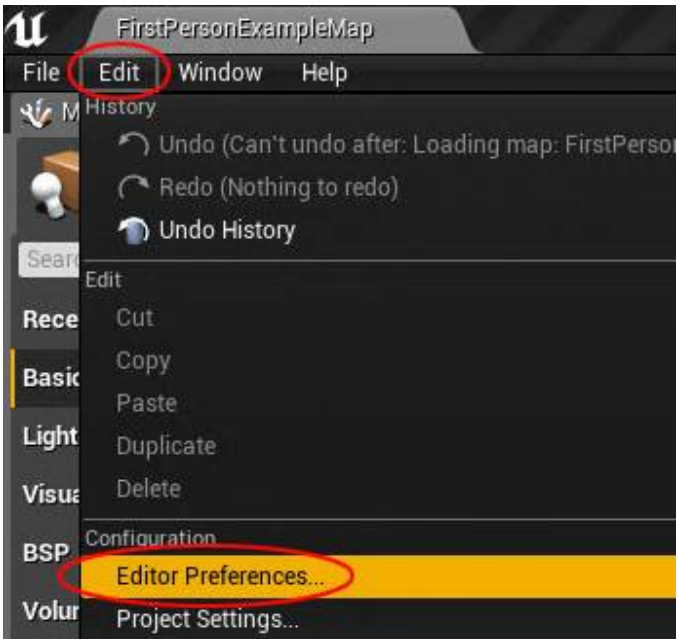
In UDK, grid size follows the power of 2 grid. This means that when you increase your grid size it goes from 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4096.



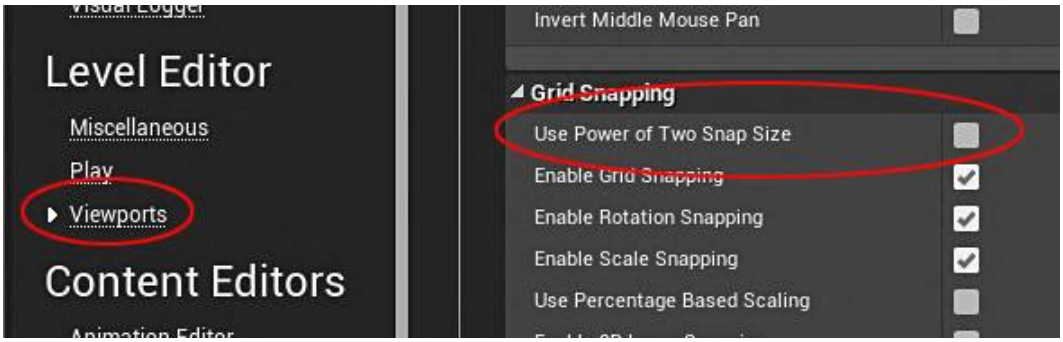
In UE4, now follows the standard decimal grid size system. Values are 1, 5, 10, 50, 100, 500, 1,000, 5,000 and 10,000. You have to set up 3d modeling application to match the grid size in Unreal Engine 4.



If you still want to use power of 2 grid in UE4, you can switch settings by going to Edit > Editor Preferences:



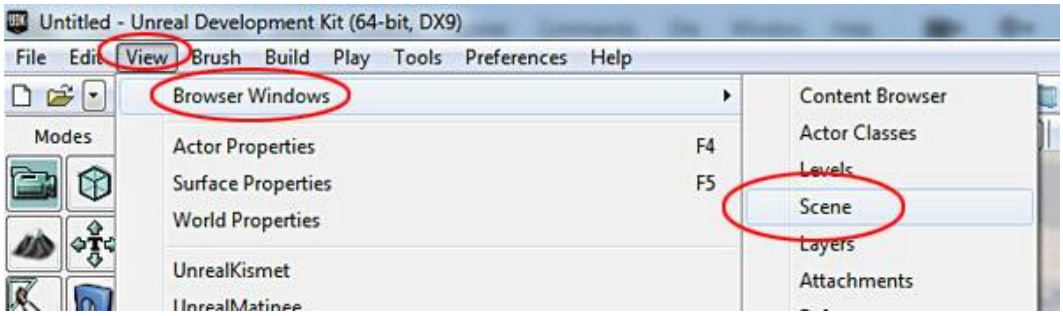
Then in Level Editor > Viewports and under the Grid Snapping option, you will see a check box to Use Power of Two Snap Size:

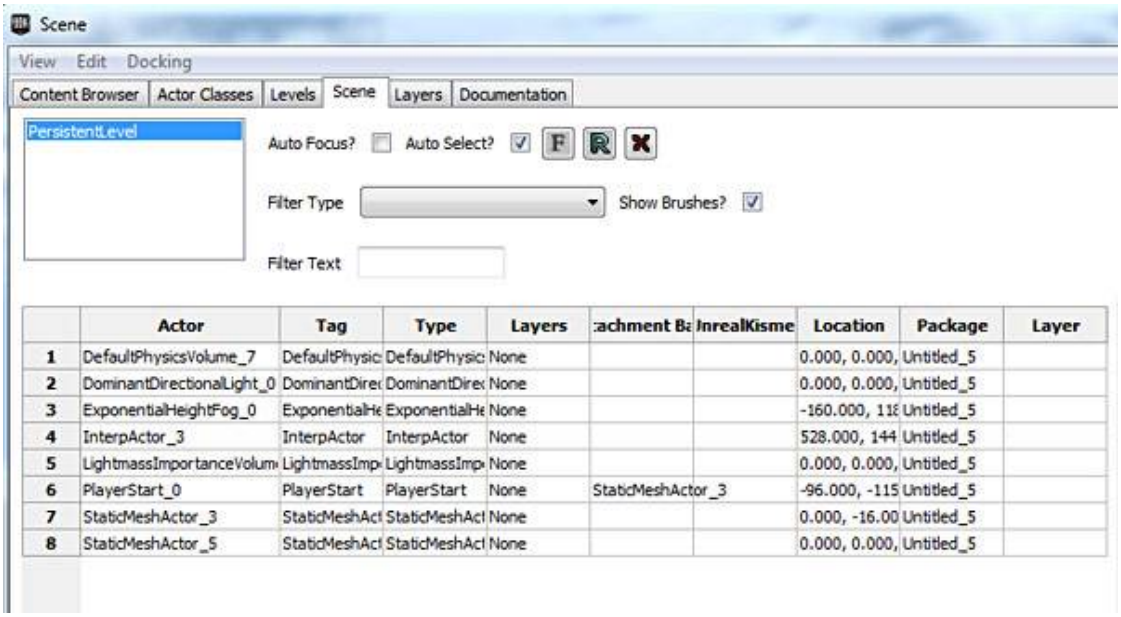


You can still use the power of 2 grid but it's best to stick with the new grid system.

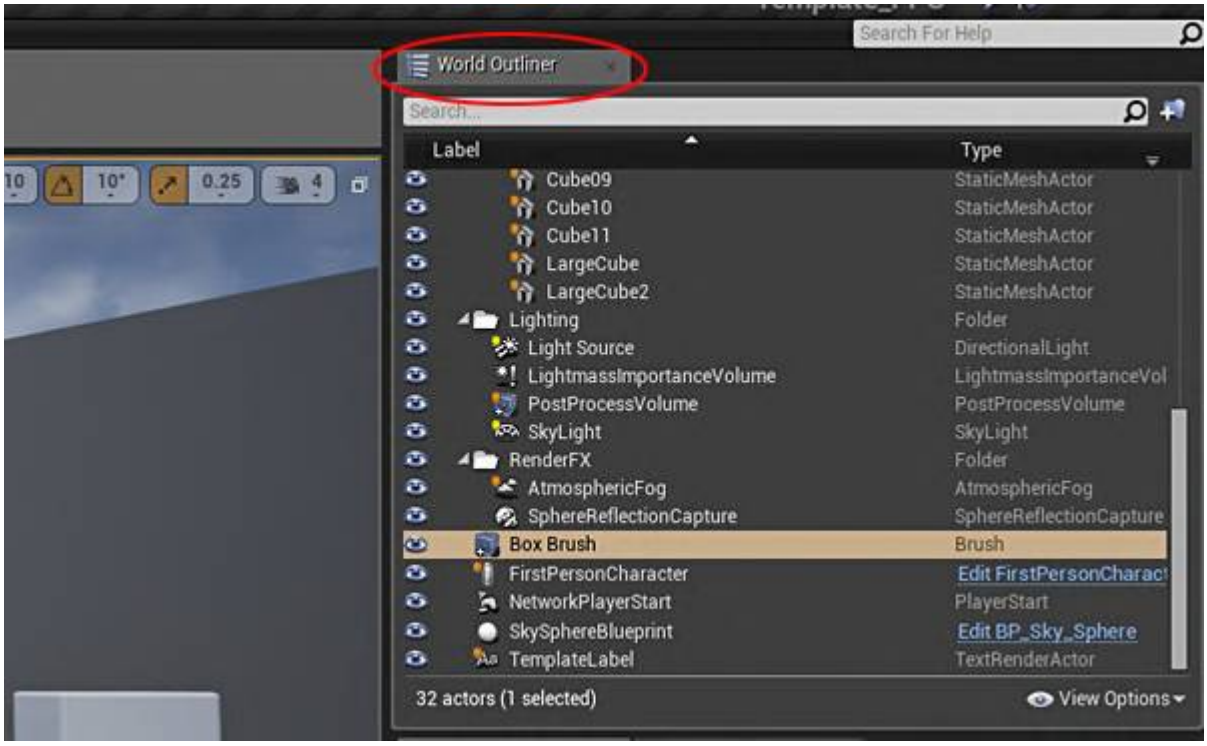
18. WORLD OUTLINER

In UDK, to see every single object placed inside your level you have to go to View > Browser Window > Scene:

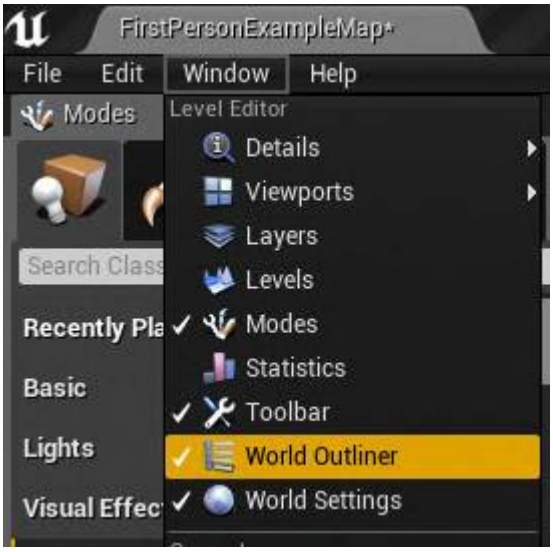




In UE4, this is now contained within its own editor panel called World Outliner and you will find this menu on upper right-hand side. This lists all of the objects inside your scene.



If you don't see the World Outliner go to Window > World Outliner:

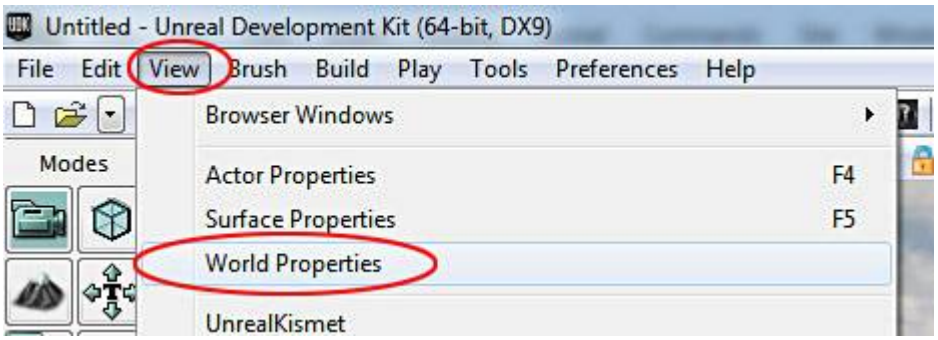


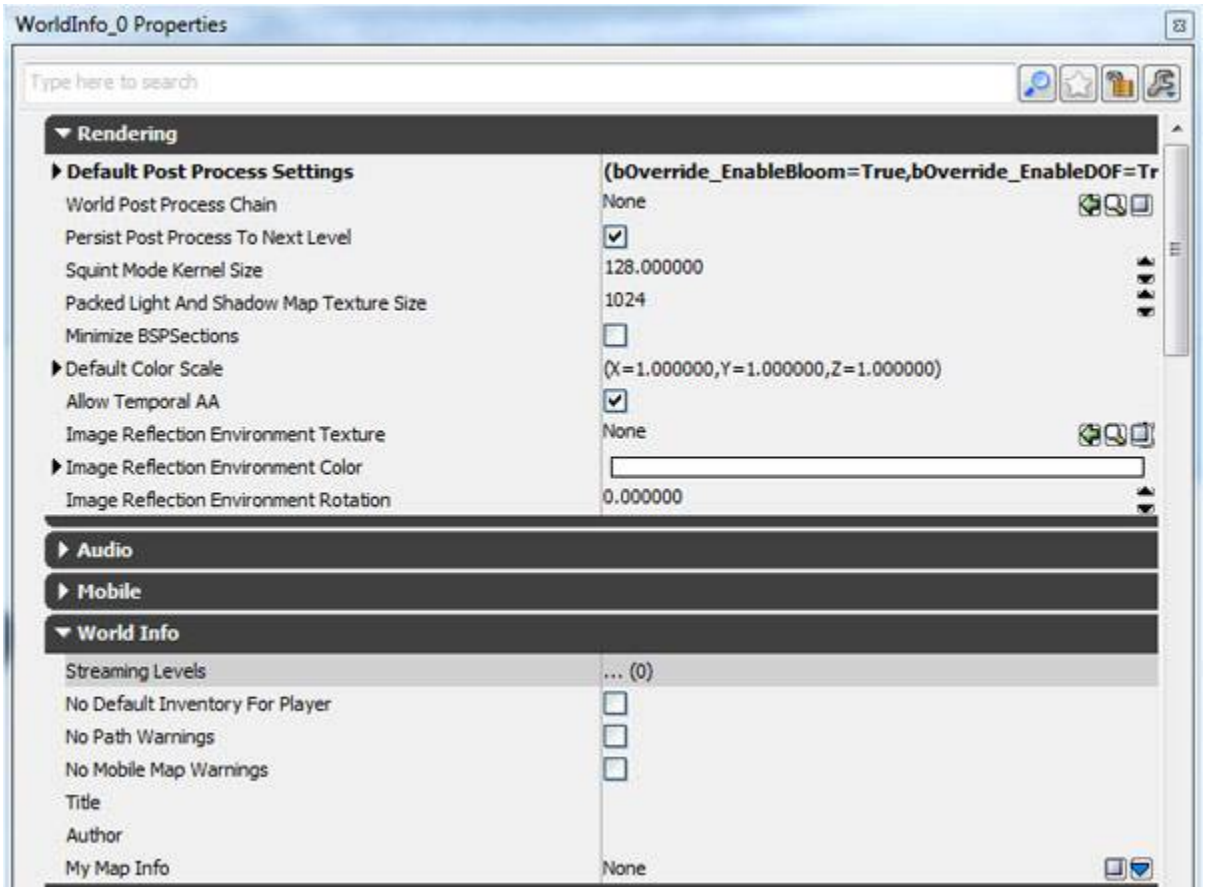
Through the World Outliner, you can select, search and organize all of the actors currently placed within your environment.

19. WORLD SETTINGS

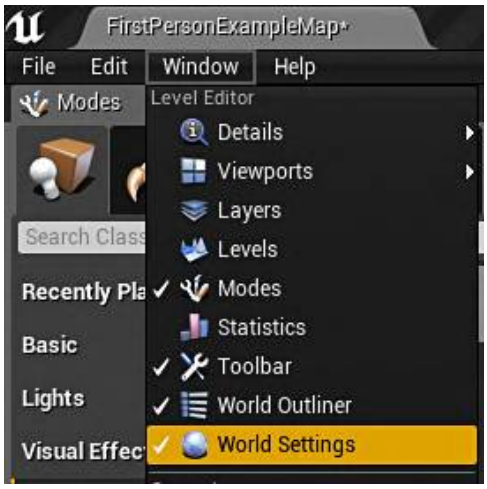
World Settings are universal properties for the entire environment. You will find settings to World, Game Mode, Physics and Lightmass.

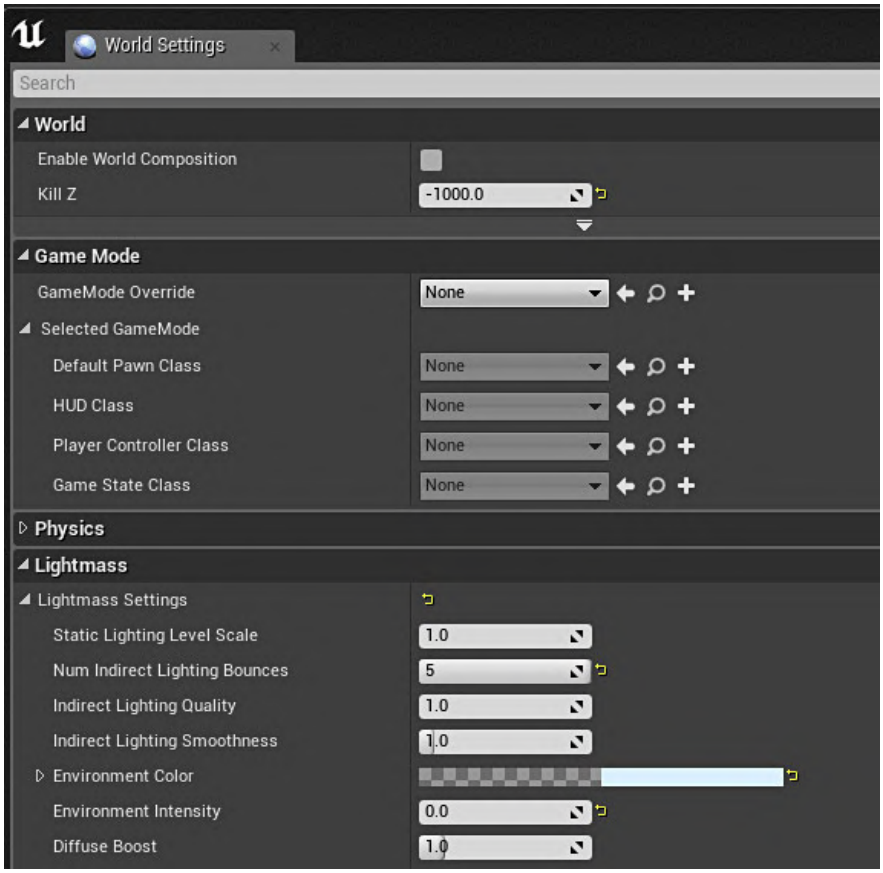
In UDK, to access World Properties, go to View > World Properties:





In UE4, World Properties are now called World Settings and you can access this menu by going to Window > World Settings:





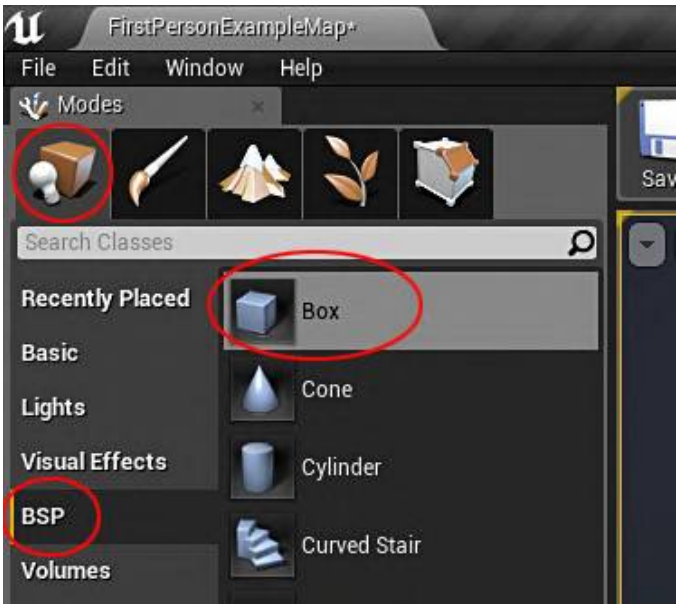
20. BSP BRUSHES

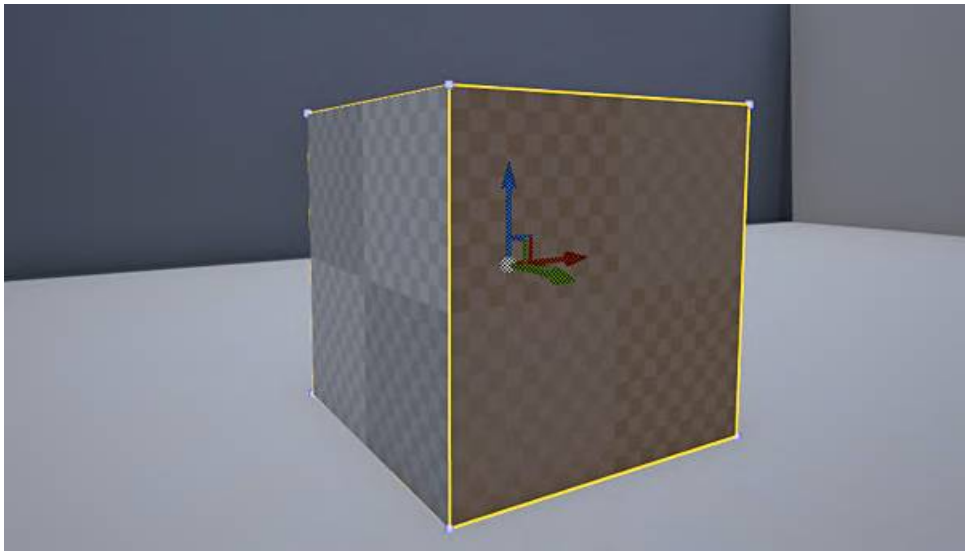
BSP brushes are extremely useful for blocking-in and prototyping a level. You can build a map with BSP brushes and have a layout to test very quickly.

In UDK, BSP process was slow and cumbersome. You had to build geometry every time you modified any BSP brush to see changes. There is a setting under Preferences to update this option to automatic BSP visualization, but it would build geometry every time you move a brush and slowed down your workflow.

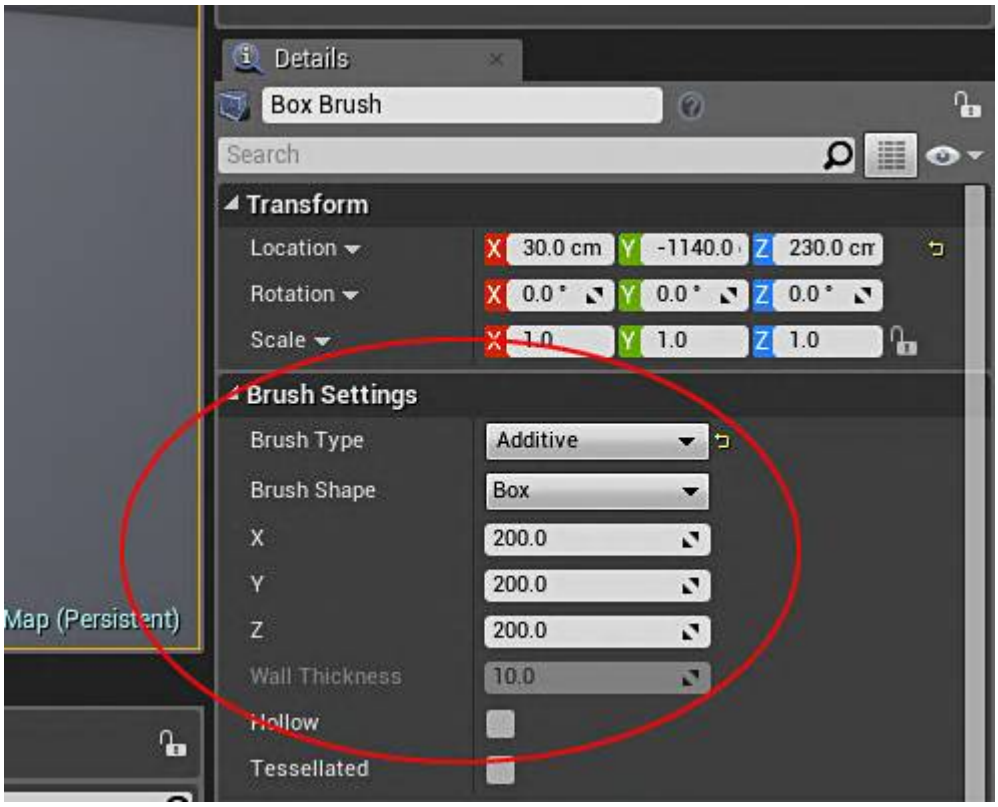
In UE4, BSP brushes process has been improved. You no longer need to be rebuilt geometry if brushes have been modified. The entire BSP process is now much faster. It's still not as streamlined as some other BSP based game engines. But it is a step in the right direction and it is much better than before.

Use Place Mode (Shift+1) to drag BSP brush into the level:

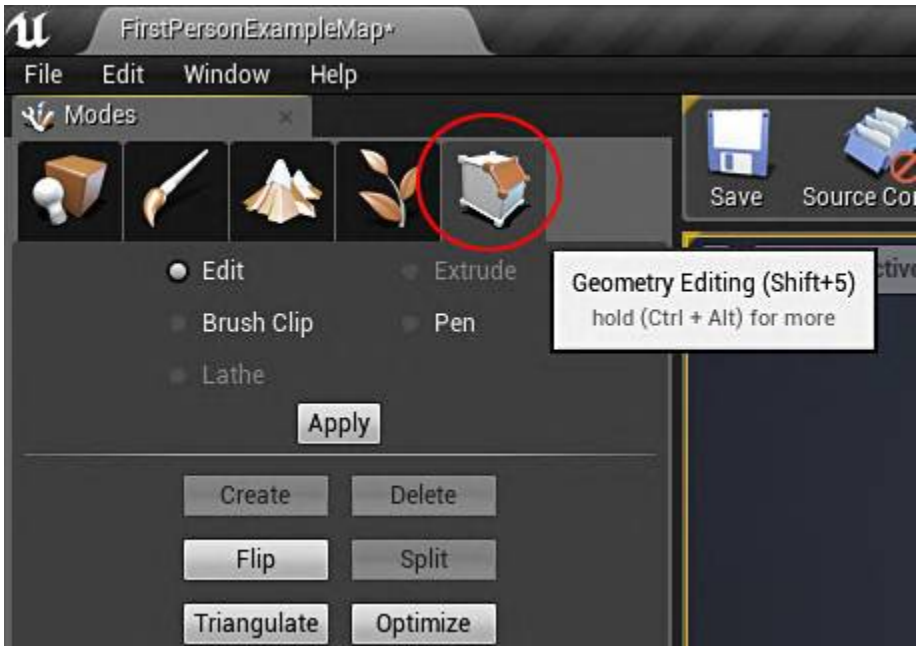


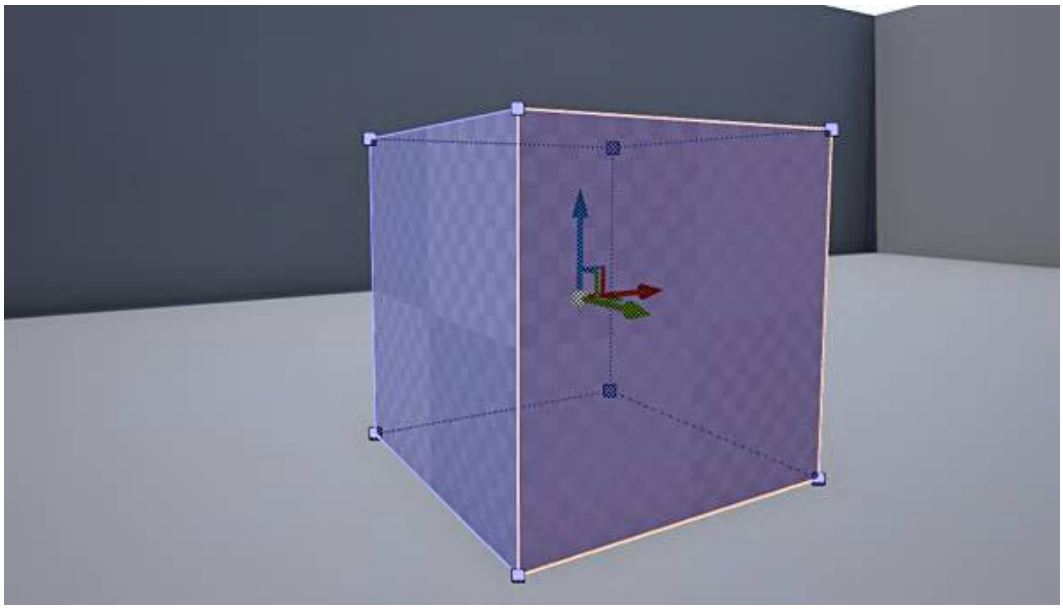


At this point you can use Details panel to input manual size values for this brush:



Or, you can use Geometry Editing Mode (Shift+5) to resize the brush via vertices, edges and faces:

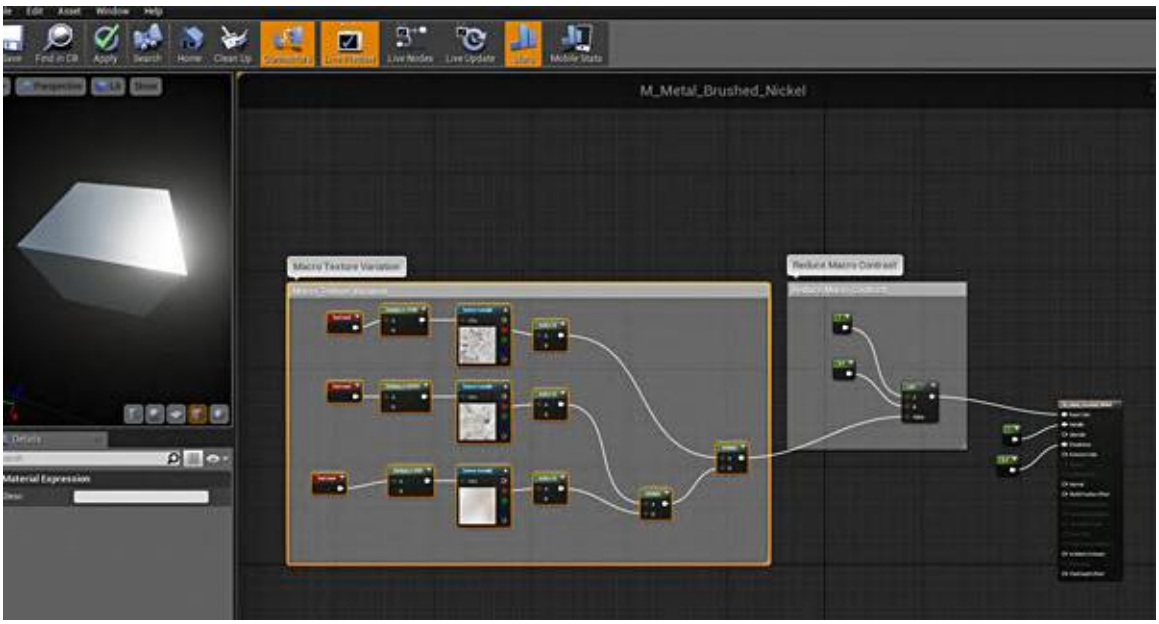




21. PBR MATERIAL CREATION

Material creation in UE4 is very different from UDK.

The material editor and how it works is similar to UDK but methodology and approach of creating materials has changed. UE4 now uses PBR or Physically Based Rendering, which offers more realistic way of rendering these materials.

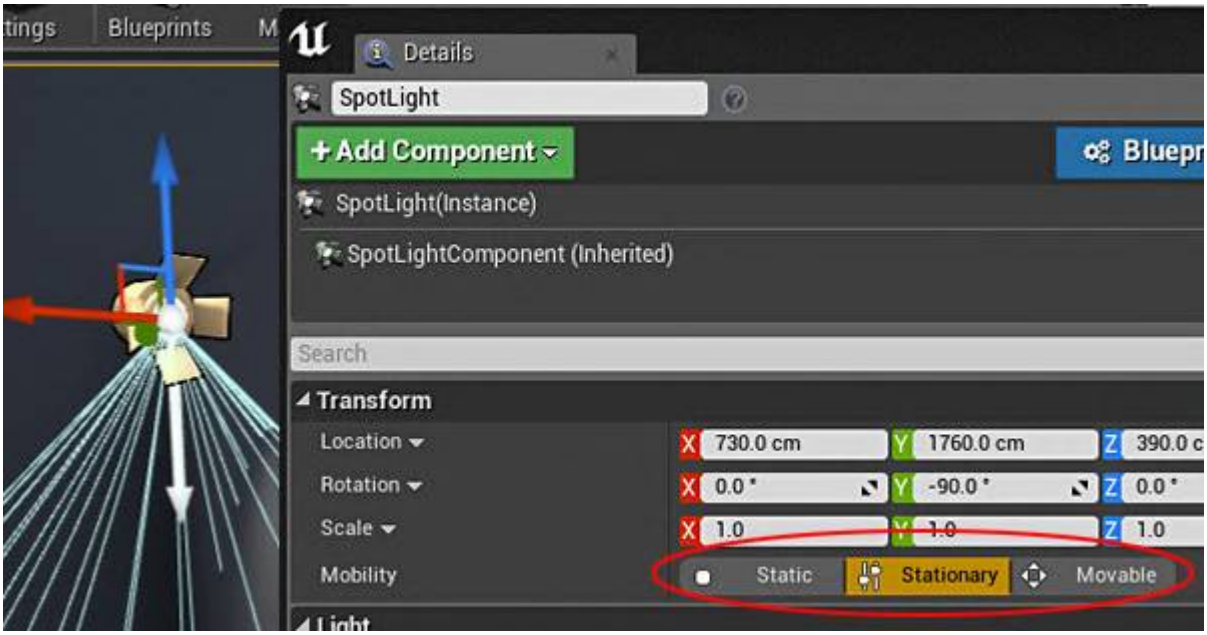


At first using PBR workflow may seem confusing. But once you learn how to construct few PBR materials, you'll find that it is actually easier than the previous workflow.

22. LIGHTING

Each light you place in UE4 will contain 3 different properties inside the Detail Panel. Static, Stationary and Moveable.

Once you insert a light, using the Details panel you can change that light to be Static, Stationary or Moveable:

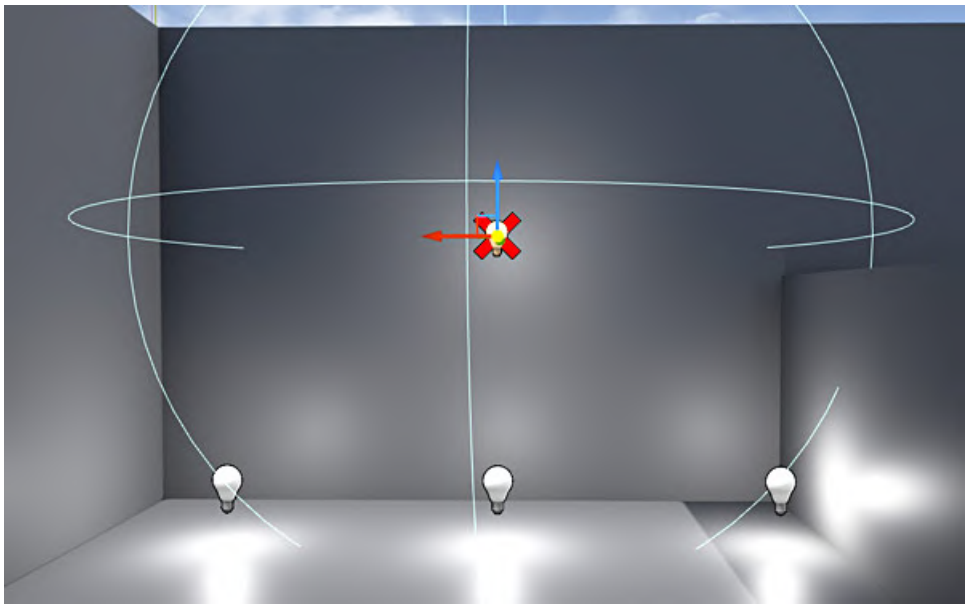


Static vs Stationary vs Movable:

- **Static**: A completely static light that has no overhead during the game
- **Stationary**: Light type that can change its color and brightness at runtime (via Blueprint) but cannot move, rotate or change influence size
- **Movable**: A fully dynamic light capable of changing all of its properties during runtime

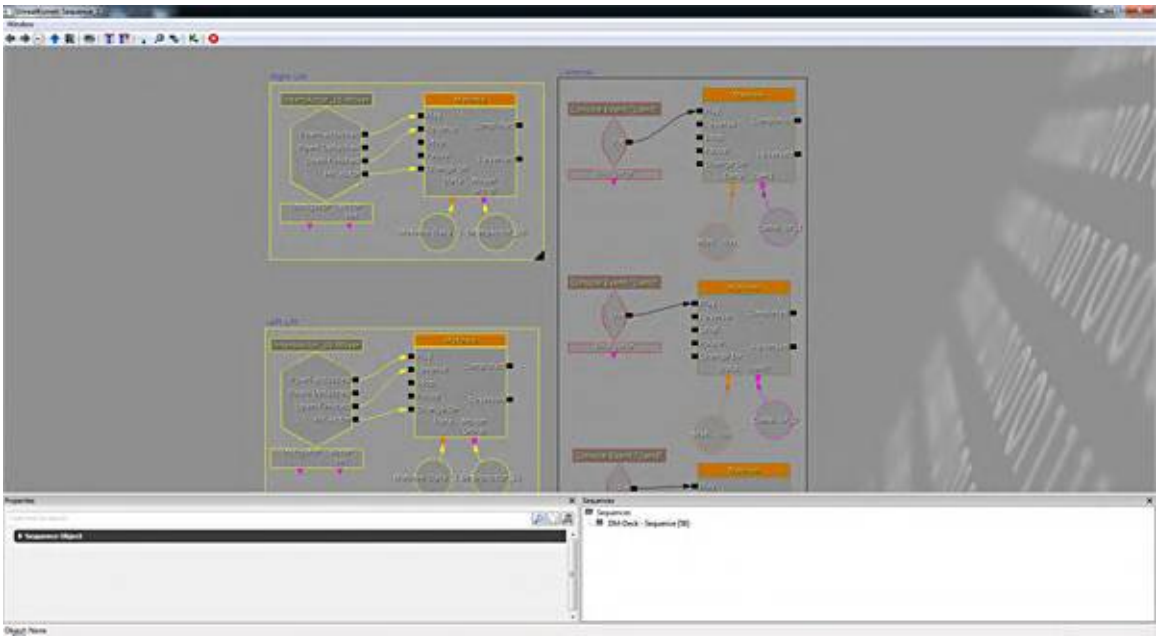
- For important lights use Stationary (default).
- For secondary fill-in lights, small area lights and far away lights use Static.
- For fully dynamic lights use Moveable.

One thing to note is that **you cannot have more than 3 overlapping Stationary lights** in an area. Meaning that you can have more than 3 Stationary lights inside your scene but the radius of influence for 3 of these Stationary lights cannot overlap at the same time:



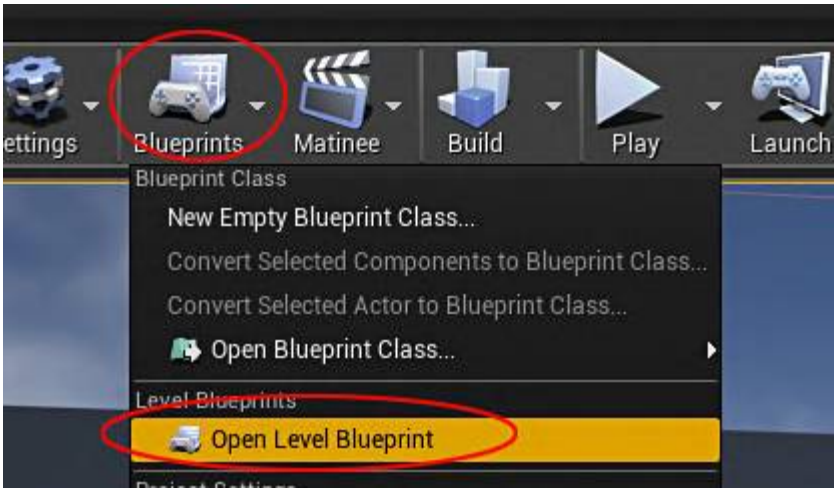
23. BLUEPRINT REPLACES KISMET

In UDK, we had Kismet, which is the visual scripting language in UDK.



In UE4, you now have Blueprint, which is a more powerful and improved system than Kismet.

Most simple way to access Blueprint is through the main toolbar:

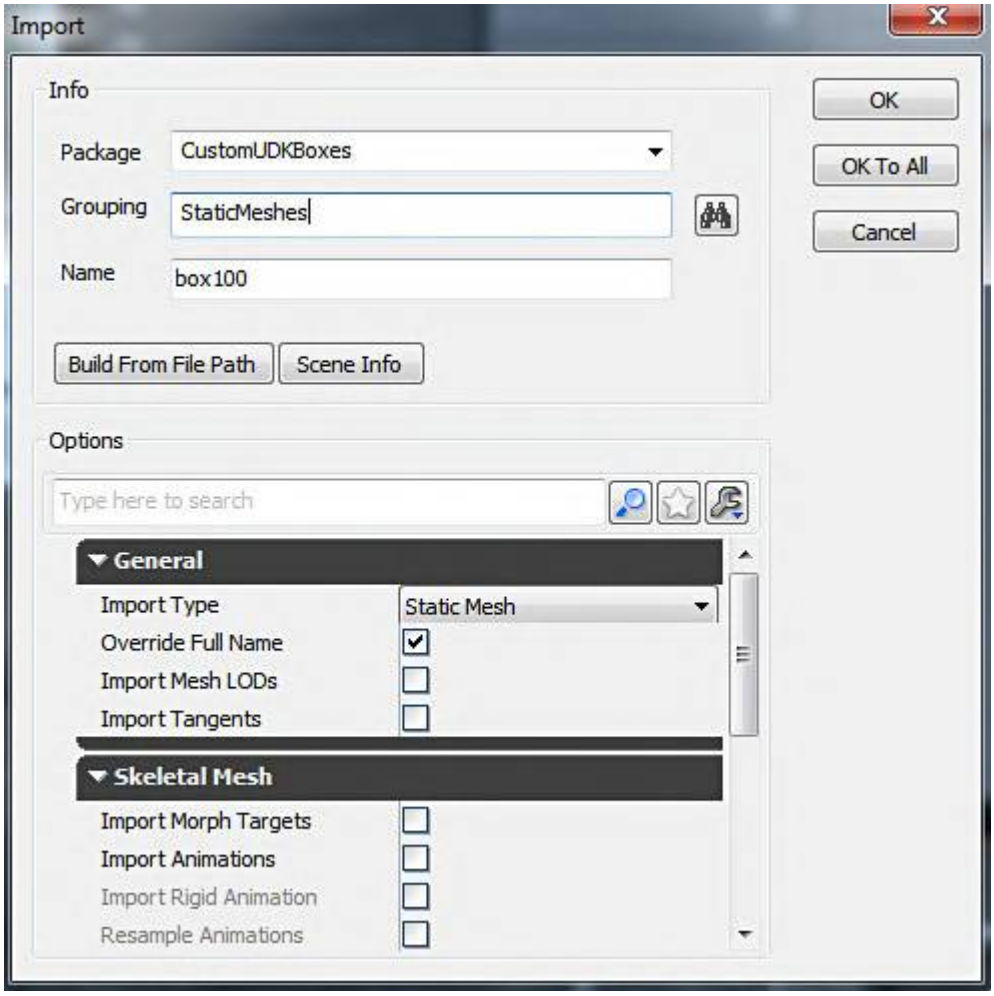


Blueprint has many improvements and functionalities. One of them is the ability to create Class Blueprints. These are self-contained items that you create to have Blueprint functionality (like a prefab). You would then place this Class Blueprint throughout the level without having to re-create it over again.

Using Blueprint you can create simple actors, components, behaviors and gameplay mechanics without having to code.

24. IMPORTING AND STORING CUSTOM ASSETS

In UDK, when you import any asset (Static Meshes, textures, materials, audio, animations etc.) you had to create a package where all of the imported content would go into.



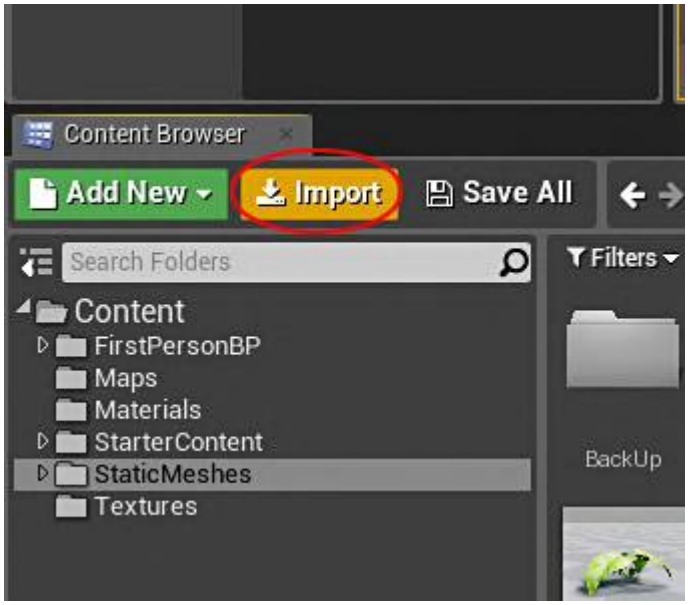
There were 2 most common file types .UDK and .UPK

- .udk = map file, this is your level that you see inside UDK
- .upk = package file, this file type contains 3d models (Static Meshes), textures, materials, sounds, animations etc that you use to place into your level

This system worked ok but I always wanted a better and more intuitive way to organize my imported content; such as a way to grab a single asset and

move it to another project without having to create a separate package to move it into.

In UE4, there are no more packages. Importing assets is very simple process. Hit import or just left-click and drag right into the Content Browser.



You now have .umap and .uasset:

- **.umap** = map or level file
- **.uasset** = individual asset file, such as a texture, a Static Mesh, a material etc.

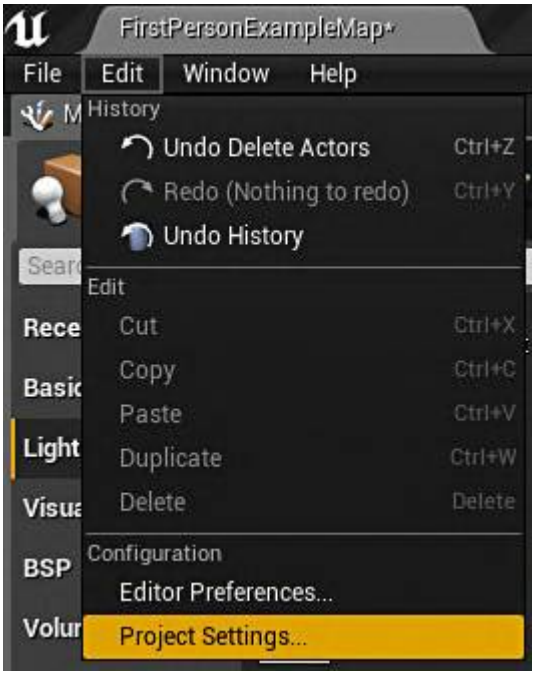
In UE4, **each project has a Content folder. This is where your game assets are stored in**. You can see the Content folder when you are inside the editor looking at the Content Browser. Anything you import for a project needs to be placed inside this Content folder. You can create sub-folders within the Content folder for better organization.

25. PROJECT SETTINGS

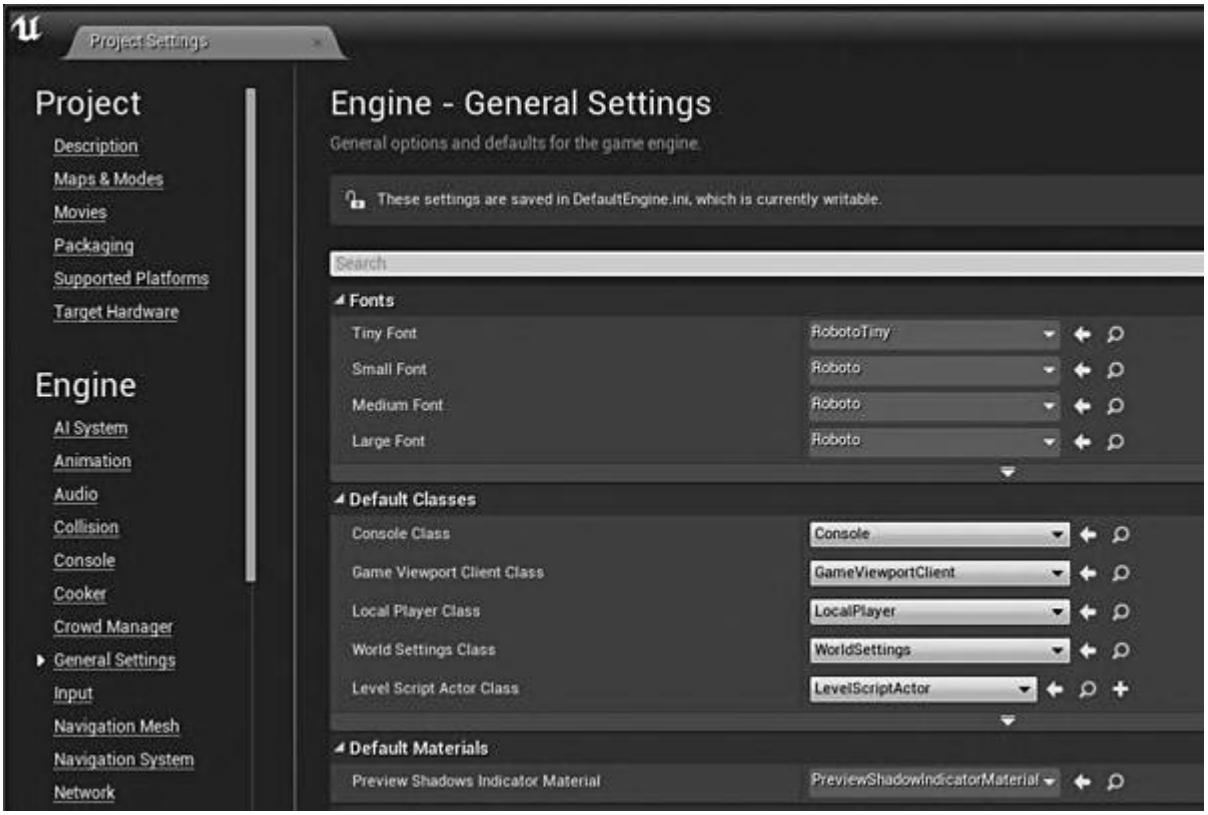
In UDK, game and level configurations were stored inside .ini configuration file. You would have to create and update this file with any changes.

In UE4, you can change your configurations inside Project Settings and you no longer need to create .ini files.

To access Project Settings go to Edit > Project Settings:



Through here you can change settings for Project, Engine and Editor:



26. CURRENT PROJECT IN UDK TO UE4

If you have a UDK project that you want to bring over to UE4. Here are some tips of what you can or cannot do:

- You can't open .upk or .udk files inside UE4
- Static Meshes and textures have to be re-imported. This should be simple if you kept your texture files and 3d modeling files (.fbx, .obj)
- Due to scale and different size use from UDK to UE4, Static Meshes used in UDK and if imported into UE4 will be smaller. You would have to scale them up in 3d modeling app or in UE4
- Materials have to be recreated following PBR workflow
- Particles Effects/Systems have to be recreated
- Kismet set-ups have to be recreated in Blueprint
- Any code has to be re-written from Unreal Script to C++
- Landscape heightmaps can be exported from UDK and imported into UE4

RECOMMENDED EXTERNAL LINKS

Epic Games - Transitioning From UE3 to UE4

Unreal Engine Wiki UE3 to UE4 Transition Guide

From Unity3D to UE4

RECOMMENDED UE4 TUTORIAL SERIES

Here are two recommended premium tutorial guides for Unreal Engine 4:

UE4 Fundamentals: this is a complete beginner guide to learning and using Unreal Engine 4. Highly recommended if you are a complete beginner with UE4.

Click here for more info...

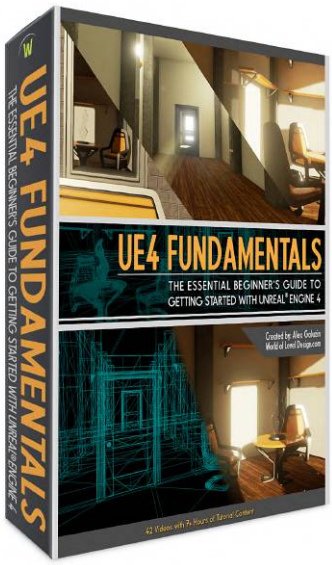
UE4 The Corridor Project: this is a bit more intermediate tutorial guide in which you construct a game environment with provided custom Static Meshes, then light it, post-process it and create a fly-through video to show it off. An in-depth guide for putting together an environment from start-to-finish.

Click here for more info...

PREMIUM VIDEO TUTORIAL GUIDES FOR UE4

For more in-depth tutorials on how to use and master Unreal Engine 4, see these premium video tutorial guides:

"UE4 FUNDAMENTALS: Essential Beginner's Guide to Getting Started with Unreal Engine 4"



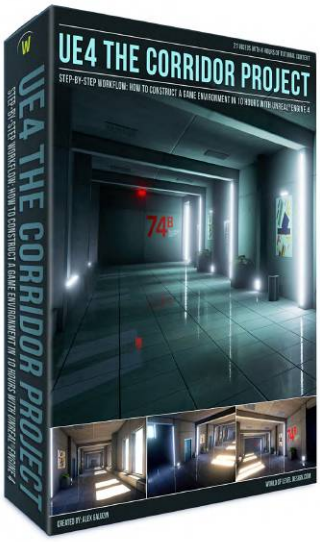
The essential beginner's guide to learning and using Unreal Engine 4. This is a complete beginner's guide to getting started with UE4.

In this guide you will get:

- 42 videos with 7 hours of tutorial content
- The Essential Beginner's Guide to Getting Started with Unreal Engine 4
- Instant Digital Download

Click here for more info...

"UE4 THE CORRIDOR PROJECT: Step-by-Step Workflow to Construct a Game Environment in Less Than 10 Hours with Unreal Engine 4"



UE4 The Corridor Project is an intermediate tutorial guide focused on constructing a game environment with provided custom Static Meshes. It's in-depth guide for putting together an environment from start-to-finish.

In this guide you will get:

- 21 videos and 4 hours of tutorial content
- Step-by-Step Workflow to Construct a Game Environment in Less Than 10 Hours with Unreal Engine 4
- Instant Digital Download

Click here for more info...